Cisco Plug-in for OpenFlow, Release 2.0.2, Configuration Guide, Cisco Nexus 7000 Series

First Published: February 15, 2016

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 IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

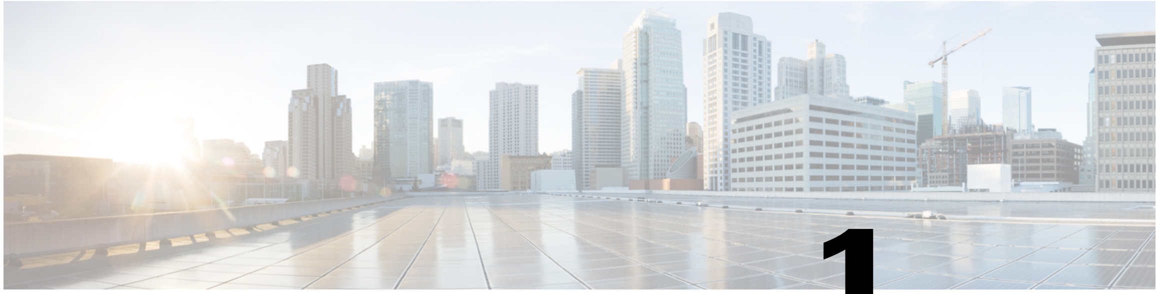
IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY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

Any 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.

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: 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. (1110R)

© Cisco Systems, Inc. All rights reserved.

CONTENTS



CHAPTER 1

New and Changed Information

New and Changed Information

The table below summarizes the new and changed features for this document and shows the releases in which each feature is supported. Your software release might not support all the features in this document. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release.

Table 1: New and Changed Cisco Plug-in for OpenFlow, Release 2.0.2 Features

Feature	Description	Changed in Release
Cisco Plug-in for OpenFlow, Release 2.0.2	Introduced in this release.	Introduced in the Cisco NX-OS Release 7.3(0)D1(1).

CHAPTER 2

Cisco Plug-in for OpenFlow

Cisco Plug-in for OpenFlow

Cisco Plug-in for OpenFlow, Release 2.0.2 provides better control over networks making them more open, programmable, and application-aware and supports the following specifications defined by the Open Networking Foundation (ONF) standards organization:

- OpenFlow Switch Specification Version 1.0.1 (Wire Protocol 0x01) (referred to as OpenFlow 1.0)
- OpenFlow Switch Specification Version 1.3.0 (Wire Protocol 0x04) (referred to as OpenFlow 1.3).

Prerequisites for Cisco Plug-in for OpenFlow

- A Cisco device and its corresponding operating system that supports the installation of Cisco Plug-in for OpenFlow.

  Note A compatibility matrix is delivered with each Cisco application. Refer to this matrix for information about the operating system releases that support features and infrastructure necessary for a particular application, such as Cisco Plug-in for OpenFlow.

- An open virtual application (OVA) package that is compatible with the device operating system and downloaded from an FTP server connected to the device.
- A controller installed on a connected server.

Table 2: Controller Support

OpenFlow Version	Supported Controllers
OpenFlow 1.0	Extensible Network Controller (XNC) 1.0, POX, or Ixia controllers
OpenFlow 1.3	Ixia or OpenDaylight

Restrictions for Cisco Plug-in for OpenFlow

- You cannot configure a bridge domain, Virtual LANs, and virtual routing and forwarding (VRF) interfaces on a Cisco Plug-in for OpenFlow logical switch.
- Cisco Plug-in for OpenFlow is not supported on default VDC.
- OpenFlow hybrid switch Integrated model is not supported. OpenFlow hybrid switch (ships-in-the-night) model is supported with physical port separation with virtual device contexts (VDCs). OpenFlow and non-OpenFlow ports must be configured on different VDCs.
- Reachability to controller via Switched Virtual Interface (SVI) is not supported.
- A routing and switching protocol must not be enabled on interfaces that are allocated to OpenFlow VDCs.
- You cannot configure more than 3000 flows in an OpenFlow VDC.

Information About Cisco Plug-in for OpenFlow

Cisco Plug-in for OpenFlow Feature Support

The following is a subset of OpenFlow 1.3 functions that are supported by Cisco Plug-in for OpenFlow.

Supported Feature	Additional Notes
OpenFlow-hybrid switch (ships-in-the-night) type is supported using OpenFlow 1.3 packet format with limitations.	OpenFlow hybrid (ships-in-the-night) hybrid model is supported with physical port seperation on virtual device contexts (VDCs). OpenFlow can be enabled on a subset of devices and ports making a part of the network OpenFlow enabled while the rest of the network continues to run using traditional forwarding principles. But the OpenFlow and non-OpenFlow ports of a device must be configured on different VDCs. OpenFlow hybrid (integrated) switch type is not supported.

Supported Feature	Additional Notes
Dedicated virtual device context (VDC) for OpenFlow	• OpenFlow can be enabled and installed on up to seven dedicated VDCs if the device has the required space. • A non default VDC must be used for OpenFlow.
Connection to up to eight controllers.	• Each Cisco Plug-in for OpenFlow VDC can connect to one controller. You can connect to up to eight controllers using seven VDCS. • Connection is via TCP. • All controllers of a VDC should be running the same OpenFlow version (1.3 or lower).
Pipelines for Cisco Plug-in for OpenFlow logical switch	• Pipelines are mandatory for the logical switch. • The logical switch supports the following pipelines: ◦ Pipeline 321 supports the L2 MAC forwarding table. ◦ Pipeline 322 supports the IPv4 and IPv6 forwarding, ARP, and L2 MAC forwarding tables.
Ethertype selector based table lookup	Ethertype of a packet decides the forwarding table and the corresponding match and action criteria. Ethertype is mandatory for pipeline 322.
Supported Interface Types	Physical interfaces and port-channel interfaces.

Supported Feature	Additional Notes
L2 Forwarding Table (Ethertype = *) (Pipeline 321)	Supported match criteria: • Source MAC address • Destination MAC address • Ethernet type (inner only) • Input port • VLAN priority code point • VLAN ID (with restrictions) **Note** If a packet contains a VLAN tag (Ethertype 0x8100), the outer Ethertype is ignored and the match is done using the VLAN ID, VLAN priority, or Inner Ethertype. Supported action criteria: • Output to multiple ports (supports up to 8 ports) • Output to controller • Set VLAN ID • Strip VLAN ID • Drop
	Supported match criteria: • Ethertype (mandatory) • IP protocol • Layer 4 source port (TCP or UDP) • Layer 4 destination port (TCP or UDP) • Input port Supported action criteria: • Output to multiple ports (supports up to 8 ports) • Punt to controller **Note** Punt to controller cannot be combined with any modify actions. • Set source MAC address (SMAC) • Set destination MAC address (DMAC) • Set VLAN ID • Strip VLAN ID • Drop

Supported Feature	Additional Notes
	Supported match criteria: • Ethertype (mandatory) • IP protocol • Layer 4 source port (TCP or UDP) • Layer 4 destination port (TCP or UDP) • Input port Supported action criteria: • Output to multiple ports (supports up to 8 ports) • Punt to controller **Note** Punt to controller cannot be combined with any modify actions. • Set source MAC address (SMAC) • Set destination MAC address (DMAC) • Set VLAN ID • Strip VLAN ID • Drop
ARP Table (Ethertype = 0x806) (Pipeline 322)	Supported match criteria: • Ethertype (mandatory) • Input port Supported action criteria: • Output to multiple ports (supports up to 8 ports) • Punt to controller • Drop
Default Action	If packets do not match flows of any of the tables above, the default action for each table is as follows: • L2 Forwarding Table-Drop • IPv4 or IPv6 Forwarding Table-Output to port on the same subnet as the destination You can also configure the default action and set it to controller if required.

Supported Feature	Additional Notes
OpenFlow v1.3 message types	The “modify state” and “queue config” message types are not supported. All other message types are supported.
Multiple actions	Flows defined on the controller must follow the guidelines below: • Multiple VLAN actions are not possible. • The flow should not have multiple rewrite actions that override one another the last action is effective. For example, strip VLAN after set VLAN or multiple set VLANs. • You cannot combine an output to port action with a punt to controller or drop action.
OpenFlow 1.3 counters	Per Port—Received Packets, Transmitted Packets, Received Bytes, Transmitted Bytes, Receive Drops, Transmit Drops, Receive Errors, Transmit Errors, Receive Frame Alignment Errors, Receive Overrun Errors, Collisions, Duration (in seconds), Duration (in nanoseconds). **Note** Per Flow and Per Table counters are not supported.

About OpenFlow

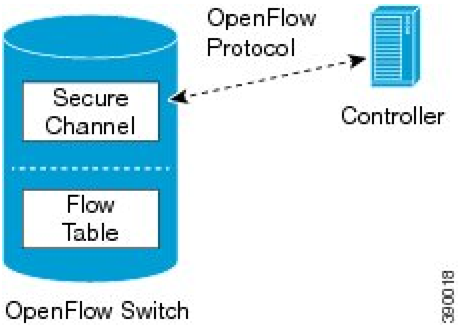
OpenFlow Switch Specification Version 1.0.1 (Wire Protocol 0x01) (referred to as OpenFlow 1.0) and OpenFlow Switch Specification Version 1.3.0 (Wire Protocol 0x04), referred to as OpenFlow 1.3, is based on the concept of an Ethernet switch, with an internal flow table and standardized interface to allow traffic flows on a device to be added or removed. OpenFlow 1.3 defines the communication channel between Cisco Plug-in for OpenFlow and controllers.

Cisco Plug-in for OpenFlow 2.0.2 refers to Cisco Plug-in for OpenFlow, Release 2.0.2.

A controller can be Extensible Network Controller (XNC) 1.0, or any controller compliant with OpenFlow 1.3.

The following figure gives an overview of the OpenFlow network.

Figure 1: OpenFlow Overview



Cisco Plug-in for OpenFlow Operation

Cisco Plug-in for OpenFlow creates OpenFlow–based TCP/IP connections to controllers for a Cisco Plug-in for OpenFlow logical switch. Cisco Plug-in for OpenFlow creates databases for a configured logical switch, OpenFlow-enabled interfaces, and flows. The logical switch database contains all the information needed to connect to a controller. The interface database contains the list of OpenFlow-enabled interfaces associated with a logical switch, and the flow database contains the list of flows on a logical switch as well as for interface that is programmed into forwarded traffic.

OpenFlow Controller Operation

OpenFlow controller (referred to as controller) controls the switch and inserts flows with a subset of OpenFlow 1.3 and 1.0 match and action criteria through Cisco Plug-in for OpenFlow logical switch. Cisco Plug-in for OpenFlow rejects all OpenFlow messages with any other action.

Cisco Plug-in for OpenFlow and Virtual Services Container

Cisco Plug-in for OpenFlow runs in an operating–system–level virtual service container on the device. The Cisco Plug-in for OpenFlow virtual service container is delivered in an open virtual application (OVA) file package (.ova). The OVA package is installed and enabled on the device through the CLI.

How to Configure Cisco Plug-in for OpenFlow

This section includes the following required and optional tasks. All tasks below require the fulfillment of the prerequisites listed in Prerequisites for Cisco Plug-in for OpenFlow, on page 3:

Configuring Physical Device Parameters

This section contains the following:

Configuring Interfaces for a Cisco Plug-in for OpenFlow Logical Switch

You must configure physical interfaces before the interfaces are added as ports of a Cisco Plug-in for OpenFlow logical switch. These interfaces are added as ports of the Cisco Plug-in for OpenFlow logical switch in the Configuring a Cisco Plug-in for OpenFlow Logical Switch , on page 16 section.

Specifying a Route to a Controller

The following tasks are used to specify a route from the device to a controller. This can be done using a physical interface (Front Panel) or a management interface.

- Physical Interface . Refer to Specifying a Route to a Controller Using a Physical Interface, on page 10.
- Management Interface. Refer to Specifying a Route to a Controller Using a Management Interface, on page 12.

The IP address of the controller is configured in the Configuring a Cisco Plug-in for OpenFlow Logical Switch , on page 16 section.

Specifying a Route to a Controller Using a Physical Interface

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type number*
3. **no switchport**
4. **ip address** *ip-address mask*
5. **exit**
6. **ip route 0.0.0.0 0.0.0.0** *next-hop*
7. **exit**
8. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface** *type number* **Example:** `Device(config)# interface Ethernet2/2`	Configures the physical interface. The interface used here should not be a Cisco Plug-in for OpenFlow ports.

	Command or Action	Purpose
Step 3	**no switchport** **Example:** `Device(config-if)# no switchport`	Configures a specified interface as a Layer 3 interface and deletes any interface configuration specific to Layer 2.
Step 4	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.0.1.4 255.255.255.0`	Configures an IP address for a specified interface.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 6	**ip route 0.0.0.0 0.0.0.0** *next-hop* **Example:** `Device(config)# ip route 0.0.0.0 0.0.0.0 10.0.1.6`	Configures a default route for packet addresses not listed in the routing table. Packets are directed toward a controller.
Step 7	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and enters privileged EXEC mode.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves the changes persistently by copying the running configuration to the startup configuration.

What to Do Next

Configure interfaces for the Cisco Plug-in for OpenFlow logical switch.

Specifying a Route to a Controller Using a Management Interface

SUMMARY STEPS

1. **configure terminal**
2. **interface mgmt** *management-interface-name number*
3. **ip address** *ip-address mask*
4. **exit**
5. **vrf context management**
6. **ip route 0.0.0.0 0.0.0.0** *next-hop*
7. **exit**
8. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface mgmt** *management-interface-name number* **Example:** `Device(config)# interface mgmt0`	Enters the management interface.
Step 3	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.0.1.4 255.255.255.0`	Configures an IP address for the interface.
Step 4	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 5	**vrf context management** **Example:** `Device(config)# vrf context management`	Configures the management Virtual routing and forwarding (VRF) instance and enters in VRF configuration mode.
Step 6	**ip route 0.0.0.0 0.0.0.0** *next-hop* **Example:** `Device(config-vrf)# ip route 0.0.0.0 0.0.0.0 10.0.1.6`	Configures a default route for packet addresses not listed in the routing table. Packets are directed toward a controller.

	Command or Action	Purpose
Step 7	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and enters privileged EXEC mode.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves the change persistently by copying the running configuration to the startup configuration.

What to Do Next

Configure interfaces for the Cisco Plug-in for OpenFlow logical switch.

Configuring Interfaces for a Cisco Plug-in for OpenFlow Logical Switch

You must configure physical interfaces before the interfaces are added as ports of a Cisco Plug-in for OpenFlow logical switch. These interfaces are added as ports of the Cisco Plug-in for OpenFlow logical switch in the Configuring a Cisco Plug-in for OpenFlow Logical Switch , on page 16 section.

Configuring a Physical Interface in Layer 2 mode

Perform the following task to add a physical interface to a Cisco Plug-in for OpenFlow logical switch in Layer 2 mode.

SUMMARY STEPS

1. **configure terminal**
2. **interface Ethernet***slot port*
3. **switchport**
4. **switchport mode trunk**
5. **mac packet-classify**
6. **switchport mode trunk allowed vlan** [*vlan-list*]
7. **no shutdown**
8. **end**
9. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**interface Ethernet***slot port* **Example:** `Device(config)# interface Ethernet2/2`	Specifies the interface for the logical switch and enters interface configuration mode.
Step 3	**switchport** **Example:** `Device(config-if)# switchport`	Specifies an interface as a Layer 2 port.
Step 4	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Specifies an interface as a trunk port. • A trunk port can carry traffic of one or more VLANs on the same physical link. (VLANs are based on the trunk-allowed VLANs list.) By default, a trunk interface carries traffic for all VLANs.
Step 5	**mac packet-classify** **Example:** `Device(config-if)# mac packet-classify`	Enables MAC packet classification on the interface.
Step 6	**switchport mode trunk allowed vlan** [*vlan-list*] **Example:** `Device(config-if)# switchport trunk allowed vlan 1-3`	Sets the list of allowed VLANs that transmit traffic from this interface in tagged format when in trunking mode.
Step 7	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Enables the interface.

Step 8	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and enters privileged EXEC mode.

Step 9	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves the change persistently by copying the running configuration to the startup configuration.

What to Do Next

Repeat these steps to configure any additional interfaces for a Cisco Plug-in for OpenFlow logical switch. Once all the interfaces are configured, install and activate Cisco Plug-in for OpenFlow.

Configuring a Physical Interface in Layer 3 mode

Perform the task below to add a physical interface to a Cisco Plug-in for OpenFlow logical switch in Layer 3 mode.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **no shutdown**
4. **end**
5. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`Device(config)# interface Ethernet1/1`
	`Device(config)# interface port-channel 101`
		Specifies the interface for the logical switch and enters interface configuration mode.

Step 3	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Enables the interface.

Step 4	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and enters privileged EXEC mode.

Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves the change persistently by copying the running configuration to the startup configuration.

What to Do Next

Repeat these steps to configure any additional interfaces for a Cisco Plug-in for OpenFlow logical switch. Once all the interfaces are configured, install and activate Cisco Plug-in for OpenFlow.

Installing and Activating Cisco Plug-in for OpenFlow

Cisco Plug-in for OpenFlow is an application that runs at the operating–system-level virtual services container on a device. Cisco Plug-in for OpenFlow is delivered in an open virtual application (OVA) package. The OVA package is installed and activated on the device through the CLI.

Before installing and activating Cisco Plug-in for OpenFlow, ensure that an OVA package compatible with the device exists on a connected FTP server. Refer to the Prerequisites for a Virtual Services Container, on page 25. A reload of the device is not essential after installing, uninstalling, or upgrading Cisco Plug-in for OpenFlow software.

To install and activate Cisco Plug-in for OpenFlow software, refer to the instructions in Installing and Activating an Application in a Virtual Services Container, on page 26, where the virtual services application argument, *virtual-services-name*, can be specified as openflow_plugin.

To uninstall and deactivate Cisco Plug-in for OpenFlow software, refer to the instructions in Deactivating and Uninstalling an Application from a Virtual Services Container, on page 28, where the virtual services application argument, *virtual-services-name*, must be the same as that specified during installation.

To upgrade Cisco Plug-in for OpenFlow software, refer to the instructions in Upgrading an Application in a Virtual Services Container, on page 29, where the virtual services application argument, *virtual-services-name*, must be the same as that specified during installation.

Once installed, configure a Cisco Plug-in for OpenFlow logical switch.

Configuring a Cisco Plug-in for OpenFlow Logical Switch

This task configures a Cisco Plug-in for OpenFlow logical switch and the IP address of a controller.

SUMMARY STEPS

1. **configure terminal**
2. **openflow**
3. **switch** *logical-switch-id*
4. **pipeline** *pipeline-id*
5. Do one of the following:
   - **of-port interface** *interface-name*
6. **protocol-version** *version-info*
7. **controller ipv4** *ip-address* [**port** *tcp-port*] [**vrf** *vrf-name*] **security**{**none** | **tls**}
8. (Optional) **logging flow-mod**
9. (Optional) **probe-interval** *probe-interval*
10. (Optional) **rate-limit packet_in** *controller-packet-rate* **burst** *maximum-packets-to-controller*
11. (Optional) **max-backoff** *backoff-timer*
12. **end**
13. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**openflow** **Example:** `Device(config)# openflow`	Enters Cisco Plug-in for OpenFlow mode.
Step 3	**switch** *logical-switch-id* **Example:** `Device(config-ofa)# switch 1`	Specifies an ID for a logical switch that is used for Layer 2 (default) switching operations and enters logical switch configuration mode. • The only logical switch ID supported is 1.
Step 4	**pipeline** *pipeline-id* **Example:**	Configures a pipeline . • This step is mandatory for a logical switch configuration. • You can view the supported pipeline values using the **show openflow hardware capabilities** command. • The valid values are from 321 and 322.

	Command or Action	Purpose
Step 5	Do one of the following: • **of-port interface** *interface-name* **Example:** For a physical interface:	Configures an Ethernet interface interface as a port of a Cisco Plug-in for OpenFlow logical switch. • Do not abbreviate the interface type. Ensure that the interface type is spelled out completely and is as shown in the examples. If the keyword is abbreviated, the interface is not configured. • The interface must be designated for the Cisco Plug-in for OpenFlow logical switch only. • The **mode openflow** configuration is added to an interface when an interface is configured as a port of Cisco Plug-in for OpenFlow. To add or remove an interface as a port of Cisco Plug-in for OpenFlow, ensure that the Cisco Plug-in for OpenFlow is activated and running to ensure the proper automatic addition and removal of the **mode openflow** configuration. To remove an interface as a port of Cisco Plug-in for OpenFlow, use the no form of this command. • Repeat this step to configure additional interfaces.
Step 6	**protocol-version** *version-info* **Example:** `Device(config-openflow-switch)# protocol-version 1.0`	Configures the protocol version. • Supported values are: ◦ **1.0—**Configures device to connect to 1.0 controllers only ◦ **1.3—**Configures device to connect to 1.3 controllers only ◦ **negotiate—**Negotiates the protocol version with the controller. Device uses 1.3 for negotiation. **Note** The default value is **negotiate**. • **drop** is the default action for both tables or pipeline 1. This can be overridden by this configuration or the controller.

Step 7	**controller ipv4** *ip-address* [**port** *tcp-port*] [**vrf** *vrf-name*] **security**{**none** | **tls**}
	Example:
	Controller in default VRF:
	`Device(config-openflow-switch)# controller ipv4 10.1.1.2 security none`
		Specifies the IPv4 address, port number, and VRF of a controller that can manage the logical switch, port number used by the controller to connect to the logical switch and the VRF of the controller.
		• If unspecified, the default VRF is used.
		• Controllers use TCP port 6653 by default.
		• You can configure up to eight controllers. Repeat this step if you need to configure additional controllers.
		• If TLS is not disabled in this step, configure TLS trustpoints in the next step.
		• You can use the **clear openflow switch 1 controller all** command to clear controller connections. This command can reset a connection after

	Command or Action	Purpose
		Transport Layer Security (TLS) certificates and keys are updated. This is not required for TCP connections. A connection to a controller is initiated for the logical switch.
Step 8	**logging flow-mod** **Example:** `Device(config-ofa-switch)# logging flow-mod`	(Optional) Enables logging of flow changes, including addition, deletion, and modification of flows. • Logging of flow changes is disabled by default. • Flow changes are logged in syslog and can be viewed using the **show logging** command. • Logging of flow changes is a CPU intensive activity and should not be enabled for networks greater than 1000 flows.
Step 9	**probe-interval** *probe-interval* **Example:** `Device(config-openflow-switch)# probe-interval 5`	(Optional) Configures the interval, in seconds, at which the controller is probed. • The default value is 5. • The range is from 5 to 65535.
Step 10	**rate-limit packet_in** *controller-packet-rate* **burst** *maximum-packets-to-controller* **Example:** `Device(config-openflow-switch)# rate-limit packet_in 1 burst 4`	(Optional) Configures the maximum packet rate of the connection to the controller and the maximum packets permitted in a burst of packets sent to the controller in a second. • The default value is zero, meaning that an indefinite packet rate and packet burst are permitted. • This rate limit is for Cisco Plug-in for OpenFlow. It is not related to the rate limit of the device (data plane) configured by COPP.
Step 11	**max-backoff** *backoff-timer* **Example:** `Device(config-openflow-switch)# max-backoff 8`	(Optional) Configures the time, in seconds, for which the device must wait before attempting to initiate a connection with the controller. • The default value is eight. • The range is from 1 to 65535.

Step 12	**end**
	Example:
	`Device(config-openflow-switch)# end`
		Exits logical switch configuration mode and enters privileged EXEC mode.

	Command or Action	Purpose
Step 13	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves the change persistently by copying the running configuration to the startup configuration.

What to Do Next

Verify Cisco Plug-in for OpenFlow.

Verifying Cisco Plug-in for OpenFlow

SUMMARY STEPS

1. **show openflow copyright**
2. **show openflow switch** *switch-id*
3. **show openflow switch** *switch-id* **controllers** [**stats**]
4. **show openflow switch** *switch-id* **ports [hidden]**
5. **show openflow switch** *switch-id* **flows** [**table-id** *table-id*][**configured | controller | default | fixed | pending | pending-del**] [**brief | summary**]
6. **show openflow switch** *switch-id* **stats**
7. **show interfaces** *type number* **counters**
8. **show logging last** *number-of-lines*
9. **show running-config | section openflow**
10. **show openflow hardware capabilities**

DETAILED STEPS

Step 1 **show openflow copyright**
Displays copyright information related to Cisco Plug-in for OpenFlow.

Example:

```
Device# show openflow copyright

Cisco Plug-in for OpenFlow
TAC support: http://www.cisco.com/tac
Copyright (c) 2013-2015 by Cisco Systems, Inc. All rights reserved.
The copyrights to certain works contained in this software are
owned by other third parties and used and distributed under
license. Certain components of this software are licensed under
the GNU General Public License (GPL) version 2.0, the GNU
Lesser General Public License (LGPL) Version 2.1, or or the GNU
Library General Public License (LGPL) Version 2. A copy of each
such license is available at
http://www.opensource.org/licenses/gpl-2.0.php and
```

```
http://www.opensource.org/licenses/lgpl-2.1.php and
http://www.gnu.org/licenses/old-licenses/lgpl-2.0.txt
```

Step 2 **show openflow switch** *switch-id*

Displays information related to Cisco Plug-in for OpenFlow logical switch.

Example:

Step 3 **show openflow switch** *switch-id* **controllers** [**stats**]

Displays information related to the connection status between an Cisco Plug-in for OpenFlow logical switch and connected controllers.

Example:

```
Device# show openflow switch 1 controllers

Logical Switch Id: 1
Total Controllers: 3
  Controller: 1
    10.1.1.2:6653
    Protocol: tcp
    VRF: default
    Connected: No
    Role: Master
    Negotiated Protocol Version: disconnected
    Last Alive Ping: N/A
    last_error:No route to host
    state:BACKOFF

  Controller: 2
    5.30.26.111:6800
    Protocol: tcp
    VRF: management
    Connected: No
    Role: Master
    Negotiated Protocol Version: disconnected
    Last Alive Ping: N/A
    last_error:Connection timed out
    state:CONNECTING
    sec_since_disconnect:14

  Controller: 3
    10.1.1.2:6653
    Protocol: tcp
    VRF: management
    Connected: No
    Role: Master
    Negotiated Protocol Version: disconnected
    Last Alive Ping: N/A
    last_error:Connection timed out
    state:CONNECTING
    sec_since_disconnect:13
```

The above sample output is displayed when controller is not yet connected.

```
Device# show openflow switch 1 controllers stats

Logical Switch Id: 1
Total Controllers: 3
  Controller: 1
    address                         :  tcp:10.1.1.2:6653
    connection attempts             :  3009
    successful connection attempts  :  0
    flow adds                       :  0
    flow mods                       :  0
    flow deletes                    :  0
    flow removals                   :  0
    flow errors                     :  0
    flow unencodable errors         :  0
    total errors                    :  0
    echo requests                   :  rx: 0, tx: 0
    echo reply                      :  rx: 0, tx: 0
    flow stats                      :  rx: 0, tx: 0
    barrier                         :  rx: 0, tx: 0
    packet-in/packet-out            :  rx: 0, tx: 0

  Controller: 2
    address                         :  tcp:5.30.26.111:6800%management
    connection attempts             :  1506
    successful connection attempts  :  0
    flow adds                       :  0
    flow mods                       :  0
    flow deletes                    :  0
    flow removals                   :  0
    flow errors                     :  0
    flow unencodable errors         :  0
    total errors                    :  0
    echo requests                   :  rx: 0, tx: 0
    echo reply                      :  rx: 0, tx: 0
    flow stats                      :  rx: 0, tx: 0
    barrier                         :  rx: 0, tx: 0
    packet-in/packet-out            :  rx: 0, tx: 0

  Controller: 3
    address                         :  tcp:10.1.1.2:6653%management
    connection attempts             :  1506
    successful connection attempts  :  0
    flow adds                       :  0
    flow mods                       :  0
    flow deletes                    :  0
    flow removals                   :  0
    flow errors                     :  0
    flow unencodable errors         :  0
    total errors                    :  0
    echo requests                   :  rx: 0, tx: 0
    echo reply                      :  rx: 0, tx: 0
    flow stats                      :  rx: 0, tx: 0
    barrier                         :  rx: 0, tx: 0
    packet-in/packet-out            :  rx: 0, tx: 0
```

Step 4 **show openflow switch** *switch-id* **ports [hidden]**

Displays the mapping between physical device interfaces and ports of an Cisco Plug-in for OpenFlow logical switch.

Example:

Step 5 **show openflow switch** *switch-id* **flows** [**table-id** *table-id*][**configured | controller | default | fixed | pending | pending-del**] [**brief | summary**]

Displays flows defined for the device by controllers.

Example:

```
Device# show openflow switch 1 flows configured

Logical Switch Id: 1
Total flows: 1

Flow: 1
  Match:
  Actions:           drop
  Priority:          0
  Table:             0
  Cookie:            0x0
  Duration:          1937.586s
  Number of packets: 0
  Number of bytes:   0
```

```
Device# show openflow switch 1 flows fixed

Logical Switch Id: 1
Total flows: 0
```

Step 6 **show openflow switch** *switch-id* **stats**
Displays send and receive statistics for each port defined for a Cisco Plug-in for OpenFlow logical switch.

Example:

Step 7 **show interfaces** *type number* **counters**
Displays send and receive statistics for the specified port defined for an Cisco Plug-in for OpenFlow logical switch.

Example:

Step 8 **show logging last** *number-of-lines*
Displays logging information of flow changes, including addition, deletion or modification of flows.

Step 9 **show running-config | section openflow**
Displays configurations made for Cisco Plug-in for OpenFlow.

Example:

Step 10 **show openflow hardware capabilities**
Displays Cisco Plug-in for OpenFlow configurations.

Example:

Configuration Examples for Cisco Plug-in for OpenFlow

Example: Specifying a Route to a Controller Using a Physical Interface

Example: Installing and Activating Cisco Plug-in for OpenFlow

Refer to *Installing and Activating an Application in a Virtual Services Container* for an example of installing and activating Cisco Plug-in for OpenFlow in a virtual services container of a device.

Additional Information for Cisco Plug-in for OpenFlow

Related Documents

Related Topic	Document Title
Cisco commands	

Standards and RFCs

Standard/RFC	Title
OpenFlow 1.3	*OpenFlow Switch Specification Version 1.3.0 (Wire Protocol 0x04).*
OpenFlow 1.0	*OpenFlow Switch Specification Version 1.0.1 (Wire Protocol 0x01).*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation and tools. Use these resources to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Cisco Plug-in for OpenFlow

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Table 3: Feature Information for Cisco Plug-in for OpenFlow

Feature Name	Releases	Feature Information
Cisco Plug-in for OpenFlow	Cisco Plug-in for OpenFlow Release 2.0.2	Cisco Plug-in for OpenFlow supports OpenFlow 1.0 and helps networks become more open, programmable, and application-aware.

CHAPTER 3

Virtual Services Container

Virtual Services Container

Prerequisites for a Virtual Services Container

- You must have a Cisco device installed with an operating system release that supports virtual services and has the needed system infrastructure required for specific applications like Cisco Plug-in for OpenFlow.

Note A compatibility matrix is delivered with each Cisco application. Refer to this matrix for information about which operating system release supports the features and infrastructure necessary for a particular application such as Cisco Plug-in for OpenFlow.

- You must download an open virtual application (OVA) package that is compatible with the device operating system, and downloaded from an FTP server connected to the device.
- You must have enough memory for installation and deployment of application. Refer to the application configuration guide for specific recommendations.

Information About Virtual Services Container

Virtual Services Containers and Applications

A virtual services container is a virtualized environment on a device. It is also referred to as a virtual machine (VM), virtual service, or container.

You can install an application within a virtual services container. The application runs in the virtual services container of the operating system of a device. The application is delivered as an open virtual application

(OVA), which is a tar file with a .ova extension. The OVA package is installed and enabled on a device through the device CLI.

Cisco Plug-in for OpenFlow is an example of an application that can be deployed within a virtual services container.

Some of the files that can be found in an OVA file are the following:

- Virtual machine definition file, in libvirt XML format, with Cisco extensions.
- Manifest file, listing the contents of a distribution. It contains the hash information for each file in the OVA package.
- Certificate file containing the signature of a manifest file. This file is used in validating the integrity of an OVA package.
- Version file, used to check compatibility with the virtualization infrastructure.

How to Configure a Virtual Services Container

This section includes the following required and optional tasks:

- Installing and Activating an Application in a Virtual Services Container, on page 26 (required)
- Deactivating and Uninstalling an Application from a Virtual Services Container, on page 28
- Upgrading an Application in a Virtual Services Container, on page 29
- Collecting General Troubleshooting Information, on page 32
- Verifying Virtual Services Container Applications , on page 33

Installing and Activating an Application in a Virtual Services Container

This task copies an open virtual application (OVA) package from an FTP file location, installs the application in a virtual services container, provisions the application, and activates it.

SUMMARY STEPS

1. **enable**
2. **copy** *from***:**//*source-directory-url destination-directory-url*
3. **virtual-service install name** *virtual-services-name* **package** *file*
4. **configure terminal**
5. **virtual-service** *virtual-services-name*
6. **activate**
7. **end**
8. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy** *from***://***source-directory-url destination-directory-url* **Example:** `Device# copy tftp://myserver.com/downloads/ofa-1.0.0-n3000-SPA-k9.ova bootflash:/ofa-1.0.0-n3000-SPA-k9.ova`	Downloads the new OVA package to the device for upgrade. Possible values are: • sftp: • tftp: • ftp: • http: • bootflash:
Step 3	**virtual-service install name** *virtual-services-name* **package** *file* **Example:** `Device# virtual-service install name openflow_agent package bootflash:/ofa-1.0.0-n3000-SPA-k9.ova`	Installs an OVA package from the specified location onto a device. Ensure that the ova file is located in the root directory of the storage device • The *virtual-services-name* defined here should be used in all occurrences of this argument in this document.
Step 4	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 5	**virtual-service** *virtual-services-name* **Example:** `Device(config)# virtual-service openflow_agent`	Configures a virtual services container and enters virtual services configuration mode. • Use the *virtual-services-name* defined during installation of the application. • Ensure that installation is complete before proceeding to the next step using the **show virtual-service list** command.
Step 6	**activate** **Example:** `Device(config-virt-serv)# activate`	Activates the installed virtual services container.

Step 7	**end**
	Example:
	`Device(config-virt-serv)# end`
		Exits virtual services configuration mode and enters privileged EXEC mode.

	Command or Action	Purpose
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

What to Do Next

You can now begin using your application.

Deactivating and Uninstalling an Application from a Virtual Services Container

(Optional) Perform this task to uninstall and deactivate an application from within a virtual services container.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **virtual-service** *virtual-services-name*
4. **no activate**
5. **no virtual-service** *virtual-services-name*
6. **end**
7. **virtual-service uninstall name** *virtual-services-name*
8. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**virtual-service** *virtual-services-name* **Example:** `Device(config)# virtual-service openflow_agent`	Enters virtual services configuration mode to configure a specified application. • Use the *virtual-services-name* defined during installation of the application.

	Command or Action	Purpose
Step 4	**no activate** **Example:** `Device(config-virt-serv)# no activate`	Disables the application.
Step 5	**no virtual-service** *virtual-services-name* **Example:** `Device(config)# no virtual-service openflow_agent`	Unprovisions the application. • Use the *virtual-services-name* defined during installation of the application. • This command is optional for all devices running Cisco IOS-XE.

Step 6	**end**
	Example:
	`Device(config-virt-serv)# end`
		Exits virtual services configuration mode and enters privileged EXEC mode.

Step 7	**virtual-service uninstall name** *virtual-services-name* **Example:** `Device# virtual-service uninstall name openflow_agent`	Uninstalls the application. • Use the *virtual-services-name* defined during installation of the application. • Run this command only after receiving a successful deactivation response from the device.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Upgrading an Application in a Virtual Services Container

(Optional) Perform this task to upgrade a virtual services container application.

Note An application upgrade may require an upgrade of the device operating system. Check the compatibility matrix of the respective application software release before upgrading it.

SUMMARY STEPS

1. **enable**
2. **copy** *from***:**//*source-directory-url destination-directory-url*
3. **configure terminal**
4. **virtual-service** *virtual-services-name*
5. **no activate**
6. **end**
7. **virtual-service upgrade name** *virtual-services-name* **package** *file*
8. **configure terminal**
9. **virtual-service** *virtual-services-name*
10. **activate**
11. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy** *from***:**//*source-directory-url destination-directory-url* **Example:** `Device# copy tftp://myserver.com/downloads/ofa-1.0.0-n3000-SPA-k9.ova bootflash:/ofa-1.0.0-n3000-SPA-k9.ova`	Downloads the new OVA package to the device for upgrade. Possible values are: • sftp: • tftp: • ftp: • http: • bootflash:
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**virtual-service** *virtual-services-name* **Example:** `Device(config)# virtual-service openflow_agent`	Enters virtual services configuration mode for configuring a specified application. • Use the *virtual-services-name* defined during installation of the application.

	Command or Action	Purpose
Step 5	**no activate** **Example:** `Device(config-virt-serv)# no activate`	Disables the application.

Step 6	**end**
	Example:
	`Device(config-virt-serv)# end`
		Exits virtual services configuration mode and enters privileged EXEC mode.

Step 7	**virtual-service upgrade name** *virtual-services-name* **package** *file* **Example:** `Device# virtual-service upgrade name openflow_agent package bootflash:/ofa-1.0.0-n3000-SPA-k9.ova`	Upgrades the application using the specified OVA file. • Use the *virtual-services-name* defined during installation of the application. • Run this command only after receiving a successful deactivation message from the device.
Step 8	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 9	**virtual-service** *virtual-services-name* **Example:** `Device(config)# virtual-service openflow_agent`	Enters virtual services configuration mode for configuration of the specified application. • Use the *virtual-services-name* defined during installation of the application.
Step 10	**activate** **Example:** `Device(config-virt-serv)# activate`	Activates the application.
Step 11	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

What to Do Next

You can now begin using your application.

Collecting General Troubleshooting Information

Information collected using the commands listed below can be sent to Cisco Technical Support for troubleshooting purposes.

SUMMARY STEPS

1. **show system sysmgr service name vman**
2. **virtual-service move name** *virtual-services-name* **[core | log] to** *destination-url*
3. **show mgmt-infra trace settings vman_trace**
4. **set trace control vman_trace buffer-size** *buffer-size*
5. **set trace control vman_trace clear [location active]**
6. **set trace vman_trace level {debug | default | err | info | warning} [location active]**

DETAILED STEPS

<table>
<tr><th></th><th>Command or Action</th><th>Purpose</th></tr>
<tr><td><b>Step 1</b></td><td>

show system sysmgr service name vman

Example:

```
Device# show system sysmgr service name vman

Service "vman" ("vman", 209):
        UUID = 0x49B, PID = 3283, SAP = 808
        State: SRV_STATE_HANDSHAKED (entered at time Tue
Mar  5 01:11:41 2013).
        Restart count: 1
        Time of last restart: Tue Mar  5 01:11:41 2013.
        The service never crashed since the last reboot.
        Tag = N/A
        Plugin ID: 0
```

</td><td>This command shows the health of the virtualization manager (VMAN) process.</td></tr>
<tr><td><b>Step 2</b></td><td>

virtual-service move name *virtual-services-name* **[core | log] to** *destination-url*

Example:

```
Device# virtual-service move name openflow_agent core to
bootflash:/
```

</td><td>Moves application log or core files to a specified destination location. This command can be used when the application running in the container has an issue (but the container is running as expected).</td></tr>
<tr><td><b>Step 3</b></td><td>

show mgmt-infra trace settings vman_trace

Example:

```
Device# show mgmt-infra trace settings vman_trace

One shot Trace Settings:

Buffer Name: vman_trace
Default Size: 262144
Current Size: 262144
Traces Dropped due to internal error: Yes
Total Entries Written: 2513
One shot mode: No
One shot and full: No
Disabled: False
```

</td><td>This command displays trace settings of a trace buffer.</td></tr>
</table>

	Command or Action	Purpose
Step 4	**set trace control vman_trace buffer-size** *buffer-size*	This command sets the trace buffer size.
Step 5	**set trace control vman_trace clear [location active]**	This command clears the trace buffer.
Step 6	**set trace vman_trace level {debug | default | err | info | warning} [location active]**	This command sets the trace level.

Verifying Virtual Services Container Applications

SUMMARY STEPS

1. **show virtual-service [global]**
2. **show virtual-service detail** [**name** *virtual-services-name*]
3. **show virtual-service list**
4. **show virtual-service storage pool list**
5. **show virtual-service storage volume list**
6. **show virtual-service version name** *virtual-services-name* **installed**
7. **show virtual-service tech-support**
8. **show virtual-service redundancy state**
9. **show virtual-service utilization name** *virtual-services-name*
10. **show virtual-service utilization statistics CPU**

DETAILED STEPS

Step 1 **show virtual-service [global]**
This command displays available memory, disk space, and CPU allocated for applications.

Example:

Step 2 **show virtual-service detail** [**name** *virtual-services-name*]
This command displays a list of resources committed to a specified application, including attached devices.

Example:

Step 3 **show virtual-service list**
This command displays an overview of resources utilized by the applications.

Example:

Step 4 **show virtual-service storage pool list**
This command displays an overview of storage locations (pools) used for virtual service containers.

Example:

```
Device# show virtual-service storage pool list

Virtual-Service storage pool list

Name                  Pool Type   Path
--------------------------------------------------------------------------
virt_strg_pool_bf_vdc_1  directory   /bootflash/virt_strg_pool_bf_vdc_1
```

Step 5 **show virtual-service storage volume list**

This command displays an overview of storage volume information for virtual service containers.

Example:

```
Device# show virtual-service storage volume list

Virtual-Service storage volume list

Name                       Capacity    In Use      Virtual-Service
--------------------------------------------------------------------------------------
_rootfs.ofa                90 MB       Yes         ofa
```

Step 6 **show virtual-service version name** *virtual-services-name* **installed**

This command displays the version of an installed application.

Example:

```
Device# show virtual-service version name openflow_agent installed

Virtual service openflow_agent installed version:
 Name : CiscoPluginForOpenFlow
 Version : 1.1.0_fc1
```

Step 7 **show virtual-service tech-support**

Displays all relevant container-based information.

Step 8 **show virtual-service redundancy state**

Example:

```
Device# show virtual-service redundancy state

Device# show virtual-service redundancy state
Virtual Service Redundancy State:


Switch No.       Role        Configure sync status      OVA sync status
----------------------------------------------------------------------------
3                Active      N/A                        N/A
```

Displays state of virtual-services.

Step 9 **show virtual-service utilization name** *virtual-services-name*

Example:

```
cat4k-openflow1#sh virtual-service utilization name openflow_agent
Virtual-Service Utilization:

CPU Utilization:
  CPU Time:  0 % (30 second average)
  CPU State: R : Running
```

```
Memory Utilization:
  Memory Allocation: 262144 Kb
  Memory Used:       19148 Kb


Storage Utilization:
  Name: _rootfs, Alias: _rootfs
    RD Bytes:    0                   WR Bytes:    0
    RD Requests: 0                   WR Requests: 0
    Errors:      0
    Capacity(1K blocks):  89243      Used(1K blocks): 66976
    Available(1K blocks): 17659      Usage: 80 %
  Name: cisco, Alias: cisco
    RD Bytes:    0                   WR Bytes:    0
    RD Requests: 0                   WR Requests: 0
    Errors:      0
    Capacity(1K blocks):  861512     Used(1K blocks): 218216
    Available(1K blocks): 643296     Usage: 26 %
  Name: /mnt/ofa, Alias: /mnt/ofa
    RD Bytes:    0                   WR Bytes:    0
    RD Requests: 0                   WR Requests: 0
    Errors:      0
    Capacity(1K blocks):  4955       Used(1K blocks): 35
    Available(1K blocks): 4664       Usage: 1  %
  Name: /cisco/core, Alias: /cisco/core
    RD Bytes:    0                   WR Bytes:    0
    RD Requests: 0                   WR Requests: 0
    Errors:      0
    Capacity(1K blocks):  138119     Used(1K blocks): 91053
    Available(1K blocks): 39935      Usage: 70 %
  Name: /tmp1, Alias: /tmp1
    RD Bytes:    0                   WR Bytes:    0
    RD Requests: 0                   WR Requests: 0
    Errors:      0
    Capacity(1K blocks):  861512     Used(1K blocks): 218216
    Available(1K blocks): 643296     Usage: 26 %
  Name: /cisco123, Alias: /cisco123
    RD Bytes:    0                   WR Bytes:    0
    RD Requests: 0                   WR Requests: 0
    Errors:      0
    Capacity(1K blocks):  856308     Used(1K blocks): 19200
    Available(1K blocks): 837108     Usage: 3  %
```

Displays virtual-services utilization information.

Step 10 **show virtual-service utilization statistics CPU**

Displays virtual service CPU utilization statistics.

Troubleshooting Virtual Services Containers

Troubleshooting Installation of Applications in a Virtual Services Container

Problem Installation of an application in a virtual services container is not successful.

Possible Cause Installation of the application may still be ongoing.

Solution Check the status of the installation using the **show virtual-service list** command. The following is sample output when the application has an Installed status.

```
Device# show virtual-service list
```

```
Virtual Service List:
Name                    Status             Package Name
-------------------------------------------------------------------
multiova                Activated          multiova-working.ova
WAAS                    Installed          ISR4451X-WAAS-5.2.0-b...
```

Possible Cause An application with the same name has already been installed.

Solution Ensure that an application of the same name has not been installed using the **show virtual-service list** command. You can verify this by referencing the Name field.

Possible Cause The target media has not been installed. Target media for various devices are given below:

- **Possible Cause** Cisco Nexus 3000 Series device—bootflash
- **Possible Cause** Cisco 4500 Series device—bootflash
- **Possible Cause** Cisco 3850 and 3650 device—flash

Solution Ensure that the target media is installed using the **show version** command.

```
Device# show 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_serie
s_home.html
Copyright (c) 2002-2013, 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.

Software
  BIOS:      version 1.2.0
  loader:    version N/A
  kickstart: version 6.0(2)U1(1)
  system:    version 6.0(2)U1(1)
  Power Sequencer Firmware:
             Module 1: version v4.4
  BIOS compile time:       08/25/2011
  kickstart image file is: bootflash:///n3000-uk9-kickstart.6.0.2.U1.0.78.bin
  kickstart compile time:  5/7/2013 12:00:00 [05/07/2013 19:45:30]
  system image file is:    bootflash:///n3000-uk9.6.0.2.U1.0.78.bin
  system compile time:     5/7/2013 12:00:00 [05/07/2013 20:54:48]

Hardware
  cisco Nexus 3048 Chassis ("48x1GE + 4x10G Supervisor")
  Intel(R) Celeron(R) CPU        P450 with 3980876 kB of memory.
  Processor Board ID FOC16434LJ2

  Device name: n3k-202-194-2
  bootflash:    2007040 kB

Kernel uptime is 0 day(s), 19 hour(s), 5 minute(s), 45 second(s)

Last reset at 132996 usecs after  Wed May  8 18:27:54 2013

  Reason: Reset Requested by CLI command reload
  System version: 6.0(2)U1(1)
  Service:

plugin
  Core Plugin, Ethernet Plugin
```

Possible Cause There is insufficient space to install an application.

Solution Ensure that sufficient space exists using the **dir** command.

```
Device# dir bootflash:

        407    May 08 21:35:52 2013  admin.rc.cli
       1332    Feb 28 16:51:27 2013  bxmnt-n3k
       3348    May 08 16:21:57 2013  config-sumana-08-may-13
    2826744    Feb 13 15:00:49 2013  dd2
    2826744    Jan 30 15:26:15 2013  dplug
   10273827    Apr 10 03:09:52 2013  gdb
     123496    Apr 10 03:12:46 2013  libexpat.so.0
       2016    Feb 28 15:18:33 2013  linux-mount-setup-n3k
    2826744    Jan 29 19:51:24 2013  lltor-dplug_md.bin
      49152    Nov 29 00:52:45 2012  lost+found/
       1903    Jan 11 16:08:49 2013  mts.log
   31884800    Apr 01 18:40:52 2013  n3000-uk9-kickstart.6.0.2.U1.0.36.bin
   31864320    Apr 08 15:53:00 2013  n3000-uk9-kickstart.6.0.2.U1.0.44.bin
   32757760    May 08 16:37:08 2013  n3000-uk9-kickstart.6.0.2.U1.0.78.bin
  232540777    Apr 04 18:24:30 2013  n3000-uk9.6.0.2.U1.0.40.bin
  232535711    Apr 08 15:51:49 2013  n3000-uk9.6.0.2.U1.0.44.bin
  232632475    May 08 16:36:35 2013  n3000-uk9.6.0.2.U1.0.78.bin
   53555200    May 08 15:37:44 2013  n3k_ofa.ova
   55101440    Feb 28 20:27:39 2013  n3k_ofa.ova-gdb
   52613120    Apr 04 18:26:55 2013  n3k_ofa.ova.port-channel2
   58675200    Feb 01 14:47:44 2013  n3k_ofa.ova1
   58675200    Feb 01 20:40:47 2013  n3k_ofa.ova31-6
    2201210    Feb 27 20:30:02 2013  of_agent
   56729600    May 08 16:41:33 2013  ofa-0.1.0_46-n3000-SSA-k9.ova
       4096    Jan 29 17:52:15 2013  onep/
       8552    Apr 04 18:10:50 2013  saveApril3
       7536    Feb 28 19:08:06 2013  saveConfigFeb28
       4096    Jan 29 00:48:00 2010  vdc_2/
       4096    Jan 29 00:48:00 2010  vdc_3/
       4096    Jan 29 00:48:00 2010  vdc_4/
       4096    May 08 18:56:52 2013  virt_strg_pool_bf_vdc_1/
       4096    Apr 09 20:24:06 2013  virtual-instance/
          0    May 08 16:51:44 2013  virtual-instance-upgrade.conf
         63    May 08 16:51:44 2013  virtual-instance.conf

Usage for bootflash://sup-local
 1558257664 bytes used
   90365952 bytes free
 1648623616 bytes total
```

Possible Cause Disk quota for container is insufficient.

Solution Ensure that disk quota available for virtual services is sufficient using the **show virtual-services global** command.

```
Device# show virtual-service global

Virtual Service Global State and Virtualization Limits:

Infrastructure version : 1.5
Total virtual services installed : 1
Total virtual services activated : 1

Machine types supported   : LXC
Machine types disabled    : KVM

Maximum VCPUs per virtual service : 1
Resource virtualization limits:
Name                         Quota     Committed     Available
--------------------------------------------------------------
system CPU (%)                   6             1             5
memory (MB)                    256           256             0
bootflash (MB)                 256           164            92
```

Possible Cause An invalid OVA package has been used for installation (Invalid package/Parsing error/Invalid machine specification error).

Solution Ensure that the OVA package copied to the device matches in size with the OVA package on the FTP server. Refer to the compatibility matrix for details or Contact Cisco Technical Support to ensure that the OVA file provided is compatible with the device operating system and not corrupted.

Possible Cause The virtual services container does not install properly due to unknown reasons.

Solution Uninstall the virtual services container. If the problem persists, collect general troubleshooting information and contact Cisco Technical Support. For more information, see Collecting General Troubleshooting Information, on page 32.

Troubleshooting Activation of Applications in a Virtual Services Container

Problem Activation of an application in a virtual services container is not successful.

Possible Cause Activation of the application may still be ongoing.

Solution Check the status of activation using the **show virtual-service list** command. The following is sample output when the application has an Activated status.

```
Device# show virtual-service list

Virtual Service List:
Name                    Status             Package Name
-------------------------------------------------------------------
WAAS                    Activated          ISR4451X-WAAS-5.2.0-b...
```

Possible Cause The virtual services container does not have sufficient resources for activation of the application.

Solution Check if the device has sufficient resources for virtualization, including memory, disk space, and CPU utilization. You can view the resource requirement for virtualization using the **show virtual-service** command.

```
Device# show virtual-service

Virtual Service Global State and Virtualization Limits:

Infrastructure version : 1.5
Total virtual services installed : 1
Total virtual services activated : 1

Machine types supported   : LXC
Machine types disabled    : KVM

Maximum VCPUs per virtual service : 1
Resource virtualization limits:
Name                         Quota     Committed     Available
--------------------------------------------------------------
system CPU (%)                   6             1             5
memory (MB)                    256           256             0
bootflash (MB)                 256           164            92
```

Possible Cause The application does not activate properly due to unknown reasons.

Solution Deactivate and uninstall the application. If the problem persists, collect general troubleshooting information and contact Cisco Technical Support. For more information, see Collecting General Troubleshooting Information, on page 32.

Troubleshooting Uninstallation of Applications in a Virtual Services Container

Problem Uninstallation of an application from the virtual services container is not successful.

Possible Cause The application being uninstalled has not deactivated completely.

Solution Check the activation status of an application using the **show virtual-service list** command. The following is sample output when the application is in the Deactivated status and can be uninstalled.

```
Device# show virtual-service list


Virtual Service List:
Name                    Status             Package Name
-------------------------------------------------------------------
WAAS                    Deactivated        ISR4451X-WAAS-5.2.0-b...
```

Possible Cause The application does not uninstall gracefully due to unknown reasons.

Solution As a last resort, delete the `virtual-instance.conf`, using the **delete** command and then reload the device.

```
Device# delete bootflash:virtual-instance.conf
Device#  reload
```

Solution If the problem persists, collect general troubleshooting information and contact Cisco Technical Support. For more information, see Collecting General Troubleshooting Information, on page 32.

Troubleshooting Deactivation of Applications in a Virtual Services Container

Problem Deactivation of an application is not successful.

Possible Cause The application being deactivated is not activated.

Solution Check the status of activation of the application using the **show virtual-service list** command. The following is sample output from a **show virtual-service list** when the application is in the Activated state and can be deactivated.

```
Device# show virtual-service list

Virtual Service List:
Name                    Status             Package Name
-------------------------------------------------------------------
oneFW                   Activated          iosxe-cx-9.0.2-hudson...
```

Possible Cause Deactivation takes a long time (5 minutes).

Solution Check if application directories are in use. Ensure that there are no shells open in the application file system directories on the device.

Possible Cause The application does not deactivate gracefully due to unknown reasons.

Solution As a last resort, uninstall the application (if you haven't done so yet) and delete the `virtual-instance.conf` configuration file, using the **delete** command and reload the device. This step deletes all applications installed in the virtual services container.

```
Device#  delete bootflash:virtual-instance.conf
Device#  reload
```

Solution If the problem persists, generate general troubleshooting information and contact Cisco Technical support. For more information, see Collecting General Troubleshooting Information, on page 32.

Configuration Examples for a Virtual Services Container

Example: Cisco Plug-in for OpenFlow Virtual Services Container Installation Configuration

```
Device# enable
Device# copy scp://myserver.com/downloads/ofa-1.0.0-n3000-SPA-k9.ova
bootflash:/ofa-1.0.0-n3000-SPA-k9.ova
Device# virtual-service install name openflow_agent package
bootflash:ofa-1.0.0-n3000-SPA-k9.ova
Device# configure terminal
Device(config)# virtual-service openflow_agent
Device(config-virt-serv)# activate
Device(config-virt-serv)# end
Device# copy running-config startup-config
```

Example: Verifying Cisco Plug-in for OpenFlow Virtual Services Container Installation Configuration

```
Device# show virtual-service list
Virtual Service List:

Name                    Status            Package Name
-----------------------------------------------------------------------
openflow_agent          Installed         ofa-1.0.0-n3000-SPA-k9.ova
```

Additional References for the Virtual Services Container

Related Documents

Related Topic	Document Title
Cisco commands	

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation and tools. Use these resources to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Virtual Services Container

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Table 4: Feature Information for the Virtual Services Container

Feature Name	Releases	Feature Information
Virtual Services Container		Cisco Plug-in for OpenFlow runs in an operating system-level virtual services container on a device. Cisco Plug-in for OpenFlow is delivered in an open virtual application (OVA). The OVA package is installed and enabled on the device through the CLI.

Glossary

application

Application installed within and hosted from a virtual ervices container on a device.

container

This is another name for virtual service container.

guest

Application instance running within a container.

host

Operating system installed on a device.

KVM

Kernel Virtual Machine. This is a virtualization infrastructure for the Linux kernel.

LxC

Linux Container. Operating system virtualization technology that shares the host kernel with the guest, but provides namespace extensions to the kernel.

logical Switch

An Cisco Plug-in for OpenFlow switch configured on a device and controlled by an external controller using flows defined on the controller.

OVA

This is an open virtual application. Software package used to install an application and related metafiles within a container. This is a tar file with a .ova extension.

physical Switch

A physical device on which Cisco Plug-in for OpenFlow application is installed and deployed.

virtual machine

This is another name for virtual service container.

virtual service

This is another name for virtual service container.

virtual services container

This is a virtualized environment on a device on which an application can be hosted. A virtualized environment on a Cisco device is called a Cisco virtual-services container.

VMAN

This is the virtualization manager. A process that manages virtual service containers and runs as a host process.