Cisco Plug-in for OpenFlowConfiguration Guide 2.0.2, Cisco Nexus 7000 Series

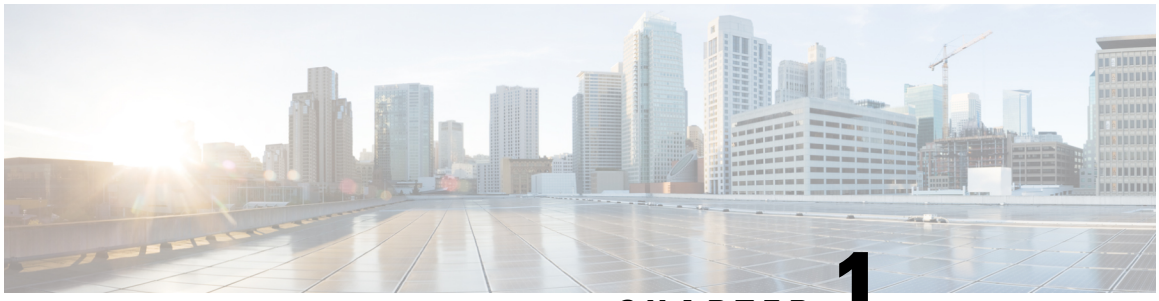
First Published: 2015-07-17

Last Modified: 2017-10-01

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

CONTENTS

CHAPTER 1



CHAPTER 1

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).

This chapter contains the following sections:

Licensing Requirements

For a complete explanation of Cisco NX-OS licensing recommendations and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

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 1: Controller Support

OpenFlow Version	Supported Controllers
OpenFlow 1.0	Extensible Network Controller (XNC) 1.0, POX, or Ixia controllers
OpenFlow 1.0	Cisco Nexus Data Broker (NDB) 2.1.0
OpenFlow 1.3	Ixia or OpenDaylight

- The required disk storage available on the device for installation and deployment of Cisco Plug-in for OpenFlow. Recommended disk space per Virtual Device Context (VDC) is 700 MB.

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.
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. • Physical interfaces in Layer 2 and Layer 3 mode assigned to the VDC must be configured as Cisco Plug-in for OpenFlow logical switch ports. • 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
IPv4 Forwarding Table (Ethertype = 0x800) (Pipeline 322)	Supported match criteria: • Ethertype (mandatory) • IP protocol • Source IP address (IPv4) • Destination IP address (IPv4) • 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
IPv6 Forwarding Table (Ethertype = 0x86DD) (Pipeline 322)	Supported match criteria: • Ethertype (mandatory) • IP protocol • Source IP address (IPv6) • Destination IP address (IPv6)) • 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 drop. You can also configure the default action and set it to controller if required.
OpenFlow v1.3 message types	The “modify state” and “queue config” message types are not supported. All other message types are supported.

Supported Feature	Additional Notes
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.

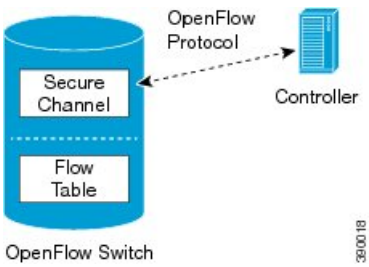
Cisco Plug-in for OpenFlow 2.0.2 refers to Cisco Plug-in for OpenFlow, Release 2.0.2.

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

In an OpenFlow network, Cisco Plug-in for OpenFlow exists on the device and controllers exist on a server, that is external to the device. Flow management and any network management are either part of a controller or accomplished through a controller. Flow management includes the addition, or removal of flows, and the handling of OpenFlow 1.3 error messages.

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 1:

- Configuring Physical Device Parameters, on page 7
- Specifying a Route to a Controller, on page 8
- Configuring Interfaces for a Cisco Plug-in for OpenFlow Logical Switch, on page 11
- Installing and Activating Cisco Plug-in for OpenFlow, on page 15
- Configuring a Cisco Plug-in for OpenFlow Logical Switch , on page 16 (optional)
- Verifying Cisco Plug-in for OpenFlow, on page 19

Configuring Physical Device Parameters

This section contains the following:

Configuring Global Variables for a Cisco Plug-in for OpenFlow Logical Switch

Before you begin

Create a non default VDC for Cisco Plug-in for OpenFlow.

SUMMARY STEPS

1. **switchto vdc** *openflow-vdc-id*

2. **configure terminal**
3. **no cdp enable**
4. **vlan** {*vlan-id* | *vlan-range*}
5. **end**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**switchto vdc** *openflow-vdc-id* **Example:** `Device# switchto vdc openflow`	Switch to the OpenFlow VDC.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no cdp enable** **Example:** `Device(config)# no cdp enable`	Disables Cisco Discovery Protocol (CDP).
Step 4	**vlan** {*vlan-id* | *vlan-range*} **Example:** `Device(config)# vlan 1-512`	Adds a VLAN or VLAN range for interfaces on the device and enters the VLAN configuration mode.
Step 5	**end** **Example:** `Device(config-vlan)# exit`	Exits VLAN configuration mode and enters privileged EXEC mode.
Step 6	**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

Specify a route to the controller.

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 9.
- Management Interface. Refer to Specifying a Route to a Controller Using a Management Interface, on page 10.

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**switchto vdc** *openflow-vdc-id* **Example:** `Device# switchto vdc openflow`	Switch to the OpenFlow VDC.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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.
Step 4	**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 5	**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 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.

	Command or Action	Purpose
Step 7	**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 8	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and enters privileged EXEC mode.
Step 9	**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. **switchto vdc** *openflow-vdc-id*
2. **configure terminal**
3. **interface mgmt** *management-interface-name number*
4. **ip address** *ip-address mask*
5. **exit**
6. **vrf context management**
7. **ip route 0.0.0.0 0.0.0.0** *next-hop*
8. **exit**
9. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**switchto vdc** *openflow-vdc-id* **Example:** `Device# switchto vdc openflow`	Switch to the OpenFlow VDC.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface mgmt** *management-interface-name number* **Example:** `Device(config)# interface mgmt0`	Enters the management interface.

	Command or Action	Purpose
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 the interface.
Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 6	**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 7	**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.
Step 8	**exit** **Example:** `Device(config)# exit`	Exits global 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

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. **switchto vdc** *openflow-vdc-id*
2. **configure terminal**
3. **interface Ethernet***slot port*

4. **switchport**
5. **switchport mode trunk**
6. **mac packet-classify**
7. **switchport mode trunk allowed vlan** [*vlan-list*]
8. **spanning-tree port type edge trunk**
9. **no shutdown**
10. **end**
11. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**switchto vdc** *openflow-vdc-id* **Example:** `Device# switchto vdc openflow`	Switch to the OpenFlow VDC.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface Ethernet***slot port* **Example:** `Device(config)# interface Ethernet2/2`	Specifies the interface for the logical switch and enters interface configuration mode.
Step 4	Required: **switchport** **Example:** `Device(config-if)# switchport`	Specifies an interface as a Layer 2 port.
Step 5	Required: **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. • This command is enabled only if the **switchport** command has been configured.
Step 6	**mac packet-classify** **Example:** `Device(config-if)# mac packet-classify`	Enables MAC packet classification on the interface.
Step 7	Required: **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.

	Command or Action	Purpose
Step 8	Required: **spanning-tree port type edge trunk** **Example:** `Device(config-if)# spanning-tree port type edge trunk`	Enables edge behavior on the trunk port. • This command is enabled only if the **switchport** command has been configured.
Step 9	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Enables the interface.
Step 10	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.
Step 11	**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. **switchto vdc** *openflow-vdc-id*
2. **configure terminal**
3. **interface** *type slot/port*
4. **no shutdown**
5. **end**
6. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**switchto vdc** *openflow-vdc-id* **Example:** `Device# switchto vdc openflow`	Switch to the OpenFlow VDC.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	

Step 3	**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 4	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Enables the interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.
Step 6	**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 Subinterface in Layer 3 mode

Perform the task below to configure one or more subinterfaces on a routed interface or on a port channel made from routed interfaces to a Cisco Plug-in for OpenFlow logical switch in Layer 3 mode.

SUMMARY STEPS

1. **switchto vdc** *openflow-vdc-id*
2. **configure terminal**
3. **interface** *type slot/port-number*
4. **encapsulation dot1Q** *vlan-id*
5. **no shutdown**
6. **end**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**switchto vdc** *openflow-vdc-id* **Example:** `Device# switchto vdc openflow`	Switch to the OpenFlow VDC.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot/port-number* **Example:** `Device(config)# interface Ethernet4/13.1`	Creates a subinterface and enters subinterface configuration mode. The valid range is from 1 to 4094.
Step 4	**encapsulation dot1Q** *vlan-id* **Example:** `Device(config-subif)# encapsulation dot1Q 501`	Configures IEEE 802.1Q VLAN encapsulation on the subinterface. The valid range is from 1 to 3967.
Step 5	**no shutdown** **Example:** `Device(config-subif)# no shutdown`	Enables the interface.
Step 6	**end** **Example:** `Device(config-subif)# end`	Exits interface configuration mode and enters privileged EXEC mode.
Step 7	**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.

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.

You must switch to the non default VDC that was created and enabled for Cisco Plug-in for OpenFlow in order to install the OVA package You can enable and install Cisco Plug-in for OpenFlow on up to three dedicated VDCs if the device has required space. slot0: of the Nexus 7000 series device must be used for kickstart and system images.

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 31. 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 32, 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 34, 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 35, 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. **switchto vdc** *openflow-vdc-id*
2. **configure terminal**
3. **openflow**
4. **switch** *logical-switch-id*
5. **pipeline** *pipeline-id*
6. Do one of the following:
   - **of-port interface** *interface-name*
   - **of-port interface** *port-channel-name*
7. **protocol-version** *version-info*
8. **controller ipv4** *ip-address* [**port** *tcp-port*] [**vrf** *vrf-name*] **security**{**none** | **tls**}
9. **default-miss** { **drop** | **controller** }
10. (Optional) **logging flow-mod**
11. (Optional) **probe-interval** *probe-interval*
12. (Optional) **rate-limit packet_in** *controller-packet-rate* **burst** *maximum-packets-to-controller*
13. (Optional) **max-backoff** *backoff-timer*
14. **end**
15. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**switchto vdc** *openflow-vdc-id* **Example:** `Device# switchto vdc openflow`	Switches to the specified VDC.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**openflow** **Example:**	Enters Cisco Plug-in for OpenFlow mode.

	Command or Action	Purpose
	`Device(config)# openflow`	
Step 4	**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 5	Required: **pipeline** *pipeline-id* **Example:** `Device(config-ofa-switch)# pipeline 321`	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.

Step 6	Do one of the following:
	• **of-port interface** *interface-name*
	• **of-port interface** *port-channel-name*
	Example:
	For a physical interface:
	`Device(config-ofa-switch)# of-port interface ethernet1/1`
	For a port-channel interface:
	`Device(config-ofa-switch)# of-port interface port-channel2`
		Configures an Ethernet interface or port-channel 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 type must be in lowercase.
		• 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 7	Required: **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.

	Command or Action	Purpose
		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 8	**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`
	Controller in management VRF:
	`Device(config-ofa-switch)# controller ipv4 10.1.1.2 vrf management 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 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 9	**default-miss** { **drop** | **controller** } **Example:** `Device(config-ofa-switch)# default-miss controller`	Configures the action to be taken for packets that do not match any of the flow defined. • **drop** is the default action for a pipeline.
Step 10	(Optional) **logging flow-mod** **Example:** `Device(config-ofa-switch)# logging flow-mod`	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 11	(Optional) **probe-interval** *probe-interval* **Example:** `Device(config-openflow-switch)# probe-interval 5`	Configures the interval, in seconds, at which the controller is probed. • The default value is 5. • The range is from 5 to 65535.

	Command or Action	Purpose
Step 12	(Optional) **rate-limit packet_in** *controller-packet-rate* **burst** *maximum-packets-to-controller* **Example:** `Device(config-openflow-switch)# rate-limit packet_in 1 burst 4`	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 13	(Optional) **max-backoff** *backoff-timer* **Example:** `Device(config-openflow-switch)# max-backoff 8`	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 14	**end** **Example:** `Device(config-openflow-switch)# end`	Exits logical switch configuration mode and enters privileged EXEC mode.
Step 15	**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:

```
Device# show openflow switch 1

Logical Switch Context
  Id: 1
  Switch type: Forwarding
  Pipeline id: 321
  Data plane: secure
  Table-Miss default: controller
  Configured protocol version: OF protocol 1.0
  Config state: no-shutdown
  Working state: enabled
  Rate limit (packet per second): 1
  Burst limit: 4
  Max backoff (sec): 8
  Probe interval (sec): 5
  TLS local trustpoint name: not configured
  TLS remote trustpoint name: not configured
  Logging flow changes: Enabled
  Stats collect interval (sec): 0
  Stats collect Max flows: 0
  Stats collect period (sec): disabled
  Minimum flow idle timeout (sec): disabled
  OFA Description:
    Manufacturer: Cisco Systems, Inc.
    Hardware: N7K-C7010 V01
    Software: 7.2(0)D1(1) of_agent 0.1
    Serial Num: TBM13384460
    DP Description: N7K_OFA_2:sw1
  OF Features:
    DPID:000100269801ccc1
    Number of tables:1
    Number of buffers:256
    Capabilities: PORT_STATS
 Controllers:
```

```
    5.30.26.111:6800, Protocol: TCP, VRF: management
    10.1.1.2:6653, Protocol: TCP, VRF: default
    10.1.1.2:6653, Protocol: TCP, VRF: management
  Interfaces:
    port-channel2
    port-channel7
    Ethernet2/2
    Ethernet2/4
    Ethernet2/5
```

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:

```
Device# show openflow switch 1 ports

Logical Switch Id: 1
Port  Interface Name    Config-State   Link-State     Features
1002  Po2               PORT_UP        LINK_DOWN      100MB-HD
1007  Po7               PORT_UP        LINK_UP        1GB-HD
5097  Eth2/5            PORT_UP        LINK_DOWN      10GB-FD
5098  Eth2/4            PORT_UP        LINK_DOWN      10GB-FD
5099  Eth2/2            PORT_UP        LINK_DOWN      10GB-FD
```

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

Logical Switch Id: 1
Total flows: 1

Flow: 1
  Match:             any
  Actions:           CONTROLLER:0
  Priority:          0
  Table:             0
  Cookie:            0x0
  Duration:          25466.484s
  Number of packets: 0
  Number of bytes:   0
```

```
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:

```
Device# show openflow switch 1 stats

Logical Switch Id: 1

Total ports: 5
  Port 1002: rx pkts=0, bytes=0, drop=0, errs=0,
             tx pkts=0, bytes=0, drop=0, errs=0,
  Port 5098: rx pkts=0, bytes=0, drop=0, errs=0,
             tx pkts=0, bytes=0, drop=0, errs=0,
  Port 1007: rx pkts=9447, bytes=3012905, drop=0, errs=0,
             tx pkts=0, bytes=0, drop=0, errs=0,
  Port 5097: rx pkts=0, bytes=0, drop=0, errs=0,
             tx pkts=0, bytes=0, drop=0, errs=0,
  Port 5099: rx pkts=0, bytes=0, drop=0, errs=0,
             tx pkts=0, bytes=0, drop=0, errs=0,
```

```
Total tables: 1
  Table 0: NXOS PLCMGR LAYER2
  Wildcards = 0x3fffff
  Max entries =   3000
  Active entries = 0
  Number of lookups = 18446744073709551615
  Number of matches = 18446744073709551615
```

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:

```
Device# show interfaces Ethernet 2/1 counters detailed

Ethernet2/1
  Rx Packets:                                     47053
  Rx Unicast Packets:                              5802
  Rx Multicast Packets:                           23908
  Rx Broadcast Packets:                           17343
  Rx Bytes:                                    12202848
  Rx Packets from 0 to 64 bytes:                  16323
  Rx Packets from 65 to 127 bytes:                14247
  Rx Packets from 256 to 511 bytes:                7053
  Rx Packets from 512 to 1023 bytes:               7674
  Rx Packets from 1024 to 1518 bytes:              1756
  Rx Trunk Packets:                                 512
  Tx Packets:                                       261
  Tx Multicast Packets:                             259
  Tx Broadcast Packets:                               2
  Tx Bytes:                                       61503
  Tx Packets from 0 to 64 bytes:                      2
  Tx Packets from 128 to 255 bytes:                 259
  Layer 3 Multicast Input Packets                 11817
  Layer 3 Multicast Input Bytes                 1057668


Device# show interfaces Ethernet 2/1 counters


--------------------------------------------------------------------------------
Port                                InOctets                      InUcastPkts
--------------------------------------------------------------------------------
Eth2/2                                     0                                0

--------------------------------------------------------------------------------
Port                              InMcastPkts                      InBcastPkts
--------------------------------------------------------------------------------
Eth2/2                                     0                                0

--------------------------------------------------------------------------------
Port                               OutOctets                     OutUcastPkts
--------------------------------------------------------------------------------
Eth2/2                                     0                                0

--------------------------------------------------------------------------------
Port                             OutMcastPkts                     OutBcastPkts
--------------------------------------------------------------------------------
Eth2/2                                     0                                0
```

Step 8 **show logging last** *number-of-lines*

Displays logging information of flow changes, including addition, deletion or modification of flows.

Example:

```
Device# show logging last 14

2013 Mar 15 19:13:05 n3k-202-194-4 %VMAN-2-ACTIVATION_STATE: Successfully activa
ted virtual service 'n3k'
2013 Mar 15 19:13:23 n3k-202-194-4 %VMAN-5-VIRT_INST: VIRTUAL SERVICE n3k LOG: E
rror: Didn't get initial config when booting up
2013 Mar 15 19:13:50 n3k-202-194-4 %VMAN-5-VIRT_INST: VIRTUAL SERVICE n3k LOG: O
VS: Flows flushed for sw1, type:cisco-l2
2013 Mar 15 19:13:54 n3k-202-194-4 %VSHD-5-VSHD_SYSLOG_CONFIG_I: Configured from
 vty by admin on console0
2013 Mar 15 19:14:09 n3k-202-194-4 %VMAN-5-VIRT_INST: VIRTUAL SERVICE n3k LOG: O
VS: Flow created: Rule: ip,dl_vlan=3 Actions: output:2,output:7
2013 Mar 15 19:14:09 n3k-202-194-4 %VMAN-5-VIRT_INST: VIRTUAL SERVICE n3k LOG: O
VS: Flow created: Rule: ip,dl_vlan=4 Actions: output:2,output:7
2013 Mar 15 19:14:09 n3k-202-194-4 %VMAN-5-VIRT_INST: VIRTUAL SERVICE n3k LOG: O
VS: Flow created: Rule: ip,dl_vlan=5 Actions: output:2,output:7
2013 Mar 15 19:14:09 n3k-202-194-4 %VMAN-5-VIRT_INST: VIRTUAL SERVICE n3k LOG: O
VS: Flow created: Rule: ip,dl_vlan=6 Actions: output:2,output:7
2013 Mar 15 19:14:09 n3k-202-194-4 %VMAN-5-VIRT_INST: VIRTUAL SERVICE n3k LOG: O
VS: Flow created: Rule: ip,dl_vlan=7 Actions: output:2,output:7
2013 Mar 15 19:14:09 n3k-202-194-4 %VMAN-5-VIRT_INST: VIRTUAL SERVICE n3k LOG: O
VS: Flow created: Rule: ip,dl_vlan=8 Actions: output:2,output:7
2013 Mar 15 19:14:09 n3k-202-194-4 %VMAN-5-VIRT_INST: VIRTUAL SERVICE n3k LOG: O
VS: Flow created: Rule: ip,dl_vlan=9 Actions: output:2,output:7
2013 Mar 15 19:14:09 n3k-202-194-4 %VMAN-5-VIRT_INST: VIRTUAL SERVICE n3k LOG: O
VS: Flow created: Rule: ip,dl_vlan=10 Actions: output:2,output:7
2013 Mar 15 19:14:09 n3k-202-194-4 %VMAN-5-VIRT_INST: VIRTUAL SERVICE n3k LOG: O
VS: Flow created: Rule: ip,dl_vlan=11 Actions: output:2,output:7
2013 Mar 15 19:14:09 n3k-202-194-4 %VMAN-5-VIRT_INST: VIRTUAL SERVICE n3k LOG: O
VS: Flow created: Rule: ip,dl_vlan=12 Actions: output:2,output:7


Device# show logging last 14

2015 Jun 26 03:18:02 N7K_OFA_2 %VMAN-5-VIRT_INST_NOTICE: VIRTUAL SERVICE ofa LOG
: Error: Calling sdna_watchdog_run
2015 Jun 26 03:18:10 N7K_OFA_2 %VMAN-5-VIRT_INST_NOTICE: VIRTUAL SERVICE ofa LOG
: last message repeated 26 times.
2015 Jun 26 03:18:10 N7K_OFA_2 %VMAN-5-VIRT_INST_NOTICE: VIRTUAL SERVICE ofa LOG
: OVS: sw1<->tcp:10.1.1.2:6653: connection failed (No route to host)
2015 Jun 26 03:18:10 N7K_OFA_2 %VMAN-5-VIRT_INST_NOTICE: VIRTUAL SERVICE ofa LOG
: Error: Calling sdna_watchdog_run
2015 Jun 26 03:18:19 N7K_OFA_2 %VMAN-5-VIRT_INST_NOTICE: VIRTUAL SERVICE ofa LOG
: last message repeated 26 times.
2015 Jun 26 03:18:19 N7K_OFA_2 %VMAN-5-VIRT_INST_NOTICE: VIRTUAL SERVICE ofa LOG
: OVS: sw1<->tcp:10.1.1.2:6653: connection failed (No route to host)
2015 Jun 26 03:18:19 N7K_OFA_2 %VMAN-5-VIRT_INST_NOTICE: VIRTUAL SERVICE ofa LOG
: Error: Calling sdna_watchdog_run
2015 Jun 26 03:18:26 N7K_OFA_2 %VMAN-5-VIRT_INST_NOTICE: VIRTUAL SERVICE ofa LOG
: last message repeated 32 times.
2015 Jun 26 03:18:26 N7K_OFA_2 %VMAN-5-VIRT_INST_NOTICE: VIRTUAL SERVICE ofa LOG
: OVS: sw1<->tcp:10.1.1.2:6653: connection failed (No route to host)
2015 Jun 26 03:18:26 N7K_OFA_2 %VMAN-5-VIRT_INST_NOTICE: VIRTUAL SERVICE ofa LOG
: Error: Calling sdna_watchdog_run
```

Step 9 **show running-config | section openflow**

Displays configurations made for Cisco Plug-in for OpenFlow.

Example:

```
Device# show running-config | section openflow

openflow
```

```
switch 1
  protocol-version 1.0
  pipeline 321
  default-miss controller
  logging flow-mod
  rate-limit packet_in 1 burst 4
  max-backoff 8
  probe-interval 5
  controller ipv4 10.1.1.2 port 6653 security none
  controller ipv4 5.30.26.111 port 6800 vrf management security none
  controller ipv4 10.1.1.2 port 6653 vrf management security none
  of-port interface ethernet2/2
  of-port interface ethernet2/4
  of-port interface ethernet2/5
  of-port interface port-channel2
  of-port interface port-channel7
```

Step 10 **show openflow hardware capabilities**

Displays Cisco Plug-in for OpenFlow configurations.

Example:

```
Device# show openflow hardware capabilities

 Pipeline ID: 321

    Pipeline Max Flows: 0

    Pipeline Default Statistics Collect Interval: 0

    Flow table ID: 0

    Max Flow Batch Size: 150
    Max Flows: 3000
    Bind Subintfs: FALSE
    Primary Table: TRUE
    Table Programmable: TRUE
    Miss Programmable: TRUE
    Number of goto tables: 0
    goto table id:
    Stats collection: Not Supported

    Match Capabilities                  Match Types
    ------------------                  -----------
    ethernet mac destination            lengthmask
    ethernet mac source                 lengthmask
    ethernet type                       optional
    VLAN ID                             optional
    VLAN priority code point            optional
    in port (virtual or physical)       optional
    wildcard all matches                optional

    Actions                     Count Limit             Order
    specified interface                 8               20
    controller                          1               20

    set vlan id                         1               10

    pop vlan tag                        1               10

    drop packet                         1               20
    Miss actions                        Count Limit             Order
    specified interface                 8               20
    controller                          1               20
```

```
  set vlan id                          1              10

  pop vlan tag                         1              10

  drop packet                          1              20



Max Flow Batch Size: 150

Statistics Max Polling Rate (flows/sec): 1024

Max Interfaces: 1000

Aggregated Statistics: NO

Pipeline ID: 322

  Pipeline Max Flows: 0

  Pipeline Default Statistics Collect Interval: 0

  Flow table ID: 0

  Max Flow Batch Size: 150
  Max Flows: 3000
  Bind Subintfs: FALSE
  Primary Table: TRUE
  Table Programmable: FALSE
  Miss Programmable: TRUE
  Number of goto tables: 4
  goto table id:      1 2 3 4
  Stats collection: Not Supported

  Match Capabilities                 Match Types
  ------------------                 -----------
  ethernet type                      mandatory

  Actions                     Count Limit            Order
  perform another lookup in the specified table      1              20



  Miss actions                       Count Limit            Order
  perform another lookup in the specified table      1              20

.
.
.
```

Configuration Examples for Cisco Plug-in for OpenFlow

Example: Configuring Global Variables for a Cisco Plug-in for OpenFlow Logical Switch

```
Device# switchto vdc openflow-
Device# configure terminal
```

```
Device(config)# no cdp enable
Device(config)# vlan 1-512
Device(config-vlan)# end
Device# copy running-config startup-config
```

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

```
Device# switchto vdc openflow
Device# configure terminal
Device(config)# interface Ethernet2/2
Device(config-if)# no switchport
Device(config-if)# ip address 10.0.1.4 255.255.255.255
Device(config-if)# exit
Device(config)# ip route 0.0.0.0 0.0.0.0 10.0.1.6
Device# copy running-config startup-config
Device(config)# exit
```

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

```
Device# switchto vdc openflow
Device# configure terminal
Device(config)# interface mgmt0
Device(config-if)# no switchport
Device(config-if)# ip address 10.0.1.4 255.255.255.255
Device(config-if)# exit
Device(config)# ip route 0.0.0.0 0.0.0.0 10.0.1.6
Device(config)# exit
Device# copy running-config startup-config
```

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.

Example: Configuring an Interface for a Cisco Plug-in for OpenFlow Logical Switch in L2 mode

```
Device# switchto vdc openflow
Device# configure terminal
Device(config)# interface ethernet1/1
Device(config-if)# switchport
Device(config-if)# switchport mode trunk
Device(config-if)# mac packet-classify
Device(config-if)# switchport trunk allowed vlan 1-3
Device(config-if)# no shutdown
Device(config-if)# exit
Device# copy running-config startup-config
```

Example: Configuring an Interface for a Cisco Plug-in for OpenFlow Logical Switch in L3 mode

```
Device# switchto vdc openflow
Device# configure terminal
Device(config)# interface ethernet1/1
Device(config)# interface port-channel 101
```

```
Device(config-if)# channel-group 2
Device(config-if)# no shutdown
Device(config-if)# exit
Device# copy running-config startup-config
```

Example: Configuring a Port-Channel Interface

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 2
Device(config-if)# switchport mode trunk
Device(config-if)# mac packet-classify
Device(config-if)# end
Device# copy running-config startup-config
```

Example: Cisco Plug-in for OpenFlow Logical Switch Configuration (Default VRF)

```
Device# switchto vdc openflow
Device# configure terminal
Device(config)# openflow
Device(config-ofa)# switch 1

! Specifies the pipeline that enables the IP Forwarding Table.

Device(config-ofa-switch)# pipeline 321
Device(config-ofa-switch)# of-port interface ethernet1/1
Device(config-ofa-switch)# of-port interface ethernet1/2
Device(config-ofa-switch)# protocol-version 1.0
Device(config-ofa-switch)# controller ipv4 10.0.1.6 security none
Device(config-ofa-switch)# default-miss controller
Device(config-ofa-switch)# probe-interval 5
Device(config-ofa-switch)# rate-limit packet_in 1 burst 4
Device(config-ofa-switch)# max-backoff 8

! Adding a port channel to the Cisco Plug-in for OpenFlow switch.

Device(config-ofa-switch)# of-port interface port-channel 2
Device(config-ofa-switch)# end
Device# copy running-config startup-config
```

Example: Configuring a Cisco Plug-in for OpenFlow Logical Switch (Management VRF)

```
Device# switchto vdc openflow
Device# configure terminal
Device(config)# openflow
Device(config-ofa)# switch 1
Device(config-ofa-switch)# pipeline 321
Device(config-ofa-switch)# controller ipv4 10.0.1.6 vrf management security none
Device(config-ofa-switch)# of-port interface ethernet1/1
Device(config-ofa-switch)# of-port interface ethernet1/2
Device(config-ofa-switch)# of-port interface port-channel 2
Device(config-ofa-switch)# end
Device# copy running-config startup-config
```

Additional Information for Cisco Plug-in for OpenFlow

Related Documents

Related Topic	Document Title
Cisco commands	Cisco Nexus 7000 Series Switches Command References

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

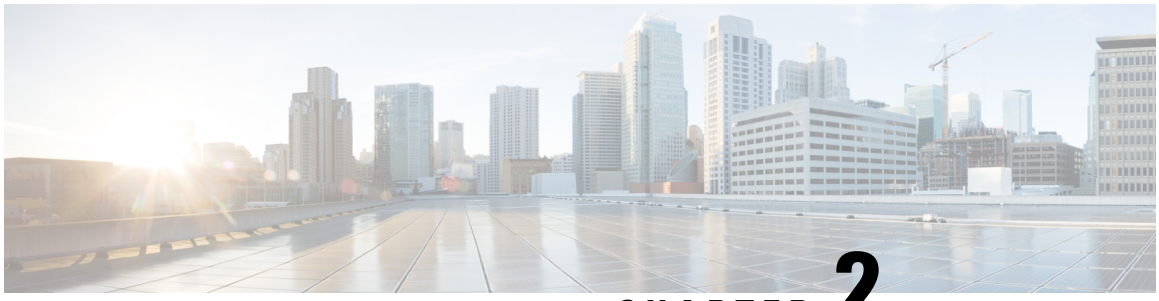
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation and tools. Use these resources to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires 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 2: 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 2

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 32 (required)
- Deactivating and Uninstalling an Application from a Virtual Services Container, on page 34
- Upgrading an Application in a Virtual Services Container, on page 35
- Collecting General Troubleshooting Information, on page 37
- Verifying Virtual Services Container Applications , on page 40

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. **switchto vdc** *openflow-vdc-id*
3. **copy** *from*://*source-directory-url destination-directory-url*
4. **virtual-service install name** *virtual-services-name* **package** *file*
5. **configure terminal**
6. **virtual-service** *virtual-services-name*
7. **activate**
8. **end**
9. **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	**switchto vdc** *openflow-vdc-id* **Example:** `Device# switchto vdc openflow`	Switches to the specified vdc.
Step 3	**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 4	**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 5	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 6	**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 7	**activate** **Example:** `Device(config-virt-serv)# activate`	Activates the installed virtual services container.
Step 8	**end** **Example:**	Exits virtual services configuration mode and enters privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-virt-serv)# end`	
Step 9	**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. **switchto vdc** *openflow-vdc-id*
3. **configure terminal**
4. **virtual-service** *virtual-services-name*
5. **no activate**
6. **no virtual-service** *virtual-services-name*
7. **end**
8. **virtual-service uninstall name** *virtual-services-name*
9. **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	**switchto vdc** *openflow-vdc-id* **Example:** `Device# switchto vdc openflow`	Switches to the specified vdc.
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 to configure 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	**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 7	**end** **Example:** `Device(config-virt-serv)# end`	Exits virtual services configuration mode and enters privileged EXEC mode.
Step 8	**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 9	**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. **switchto vdc** *openflow-vdc-id*
3. **copy** *from***://***source-directory-url destination-directory-url*
4. **configure terminal**
5. **virtual-service** *virtual-services-name*
6. **no activate**
7. **end**
8. **virtual-service upgrade name** *virtual-services-name* **package** *file*

9. **configure terminal**
10. **virtual-service** *virtual-services-name*
11. **activate**
12. **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	**switchto vdc** *openflow-vdc-id* **Example:** `Device# switchto vdc openflow`	Switches to the specified vdc.
Step 3	**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 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`	Enters virtual services configuration mode for configuring a specified application. • Use the *virtual-services-name* defined during installation of the application.
Step 6	**no activate** **Example:** `Device(config-virt-serv)# no activate`	Disables the application.
Step 7	**end** **Example:** `Device(config-virt-serv)# end`	Exits virtual services configuration mode and enters privileged EXEC mode.
Step 8	**virtual-service upgrade name** *virtual-services-name* **package** *file*	Upgrades the application using the specified OVA file.

	Command or Action	Purpose
	Example: `Device# virtual-service upgrade name openflow_agent package bootflash:/ofa-1.0.0-n3000-SPA-k9.ova`	• 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 9	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 10	**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 11	**activate** **Example:** `Device(config-virt-serv)# activate`	Activates the application.
Step 12	**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. **show system virtual-service event-history debug**
3. **show logging level virtual-service**
4. **show logging last** *number-of_-ines* | **include VMAN**
5. **virtual-service move name** *virtual-services-name* **[core | log] to** *destination-url*
6. **show mgmt-infra trace settings vman_trace**
7. **set trace control vman_trace buffer-size** *buffer-size*
8. **set trace control vman_trace clear [location active]**
9. **set trace vman_trace level {debug | default | err | info | warning} [location active]**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**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`
		This command shows the health of the virtualization manager (VMAN) process.

Step 2	**show system virtual-service event-history debug**
	Example:
	`Device# show system virtual-service event-history debug`
	
	`1) Event:E_VMAN_MSG, length:42, at 373061 usecs after Thu May 9 20:03:45 2013`
	`(debug): Queueing unprocessed MTS message`
	
	`2) Event:E_VMAN_MSG, length:42, at 92367 usecs after Thu May 9 19:53:29 2013`
	`(debug): Queueing unprocessed MTS message`
	
	`3) Event:E_VMAN_MSG, length:42, at 300136 usecs after Thu May 9 19:53:21 2013`
	`(debug): Queueing unprocessed MTS message`
	
	`4) Event:E_VMAN_MSG, length:42, at 56305 usecs after Thu May 9 19:51:22 2013`
	`(debug): Queueing unprocessed MTS message`
	
	`5) Event:E_VMAN_MSG, length:91, at 209708 usecs after Thu May 9 09:57:23 2013`
	`(debug): Storage(MB): pools(265) committed(275) quota(600) credit(0), libvirt is connected`
	
	`6) Event:E_VMAN_MSG, length:70, at 209700 usecs after Thu May 9 09:57:23 2013`
	`(debug): Disk space committed by pool virt_strg_pool_bf_vdc_1 = 275MB`	

	Command or Action	Purpose

Step 3	**show logging level virtual-service**
	Example:
	`Device# show logging level virtual-service`
	
	`Facility Default Severity Current Session Severity`
	`-------- ---------- ----------`
	`virtual-service 5 5`
	
	`0(emergencies) 1(alerts) 2(critical)`
	`3(errors) 4(warnings)`
	`5(notifications)`
	`6(information) 7(debugging)`
		This command contains information related to the VMAN configuration.

Step 4	**show logging last** *number-of_-ines* | **include VMAN**
	Example:
	`Device# show logging last 100 | include VMAN`
	
	`2013 May 8 18:31:26 n3k-202-194-2`
	`%VMAN-2-INSTALL_STATE: Successfully installed virtual service 'openflow_agent'`
	`2013 May 8 18:57:15 n3k-202-194-2`
	`%VMAN-2-ACTIVATION_STATE: Successfully activated virtual service 'openflow_agent'`
	`2013 May 8 18:57:15 n3k-202-194-2`
	`%VMAN-5-VIRT_INST: LOG FROM VIRTUAL SERVICE n3k: OVS: sw1<->tcp:10.86.201.161:6633%management: connected`
	`2013 May 9 14:58:47 n3k-202-194-2`
	`%VMAN-5-VIRT_INST: LOG FROM VIRTUAL SERVICE n3k: OVS: sw1<->tcp:10.44.94.173:6633%management: connected`
	`2013 May 9 15:00:05 n3k-202-194-2`
	`%VMAN-5-VIRT_INST: LOG FROM VIRTUAL SERVICE n3k: OVS: sw1<->tcp:10.168.1.31:7777: connected`
		This command shows the VMAN logging configuration and contents of log files.

Step 5	**virtual-service move name** *virtual-services-name* **[core | log] to** *destination-url* **Example:** `Device# virtual-service move name openflow_agent core to bootflash:/`	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).

Step 6	**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`
		This command displays trace settings of a trace buffer.

	Command or Action	Purpose

	`Total Entries Written: 2513`
	`One shot mode: No`
	`One shot and full: No`
	`Disabled: False`	

Step 7	**set trace control vman_trace buffer-size** *buffer-size*	This command sets the trace buffer size.
Step 8	**set trace control vman_trace clear [location active]**	This command clears the trace buffer.
Step 9	**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:

```
Device# show virtual-service

Virtual Service Global State and Virtualization Limits:

Infrastructure version : 1.3
Total virtual services installed : 1
Total virtual services activated : 1

Maximum memory for virtualization : 768 MB
Maximum HDD storage for virtualization : 0 MB
Maximum bootflash storage for virtualization : 600 MB
Maximum system CPU : 6%
Maximum VCPUs per virtual service : 1

Committed memory          : 700 MB
Committed disk storage    : 275 MB
Committed system CPU      : 1%
```

```
Available memory          : 68 MB
Available disk storage    : 165 MB
Available system CPU      : 5%
Machine types supported   : LXC
Machine types disabled    : KVM
```

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:

```
Device# show virtual-service detail name openflow_agent

Virtual service openflow_agent detail
  State                : Activated
  Package information
    Name               : ofa-0.1.0_46-n3000-SSA-k9.ova
    Path               : bootflash:/ofa-0.1.0_46-n3000-SSA-k9.ova
    Application
      Name             : CiscoPluginForOpenFlow
      Installed version : 1.1.0_fc1
      Description      : Cisco Plug-in for OpenFlow
    Signing
      Key type         : Cisco release key
      Method           : SHA-1
    Licensing
      Name             : None
      Version          : None
  Resource reservation
    Disk               : 275 MB
    Memory             : 700 MB
    CPU                : 1% system CPU

  Attached devices
    Type              Name        Alias
    ---------------------------------------------
    Watchdog          watchdog-226.0
    Serial/Trace                  serial3
    Serial/Syslog                 serial2
    Serial/aux
    Serial/shell
    Disk              /mnt/core
    Disk              /mnt/ofa
    Disk              _rootfs
```

Step 3 **show virtual-service list**

This command displays an overview of resources utilized by the applications.

Example:

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

Name                      Status             Package Name
------------------------------------------------------------------------
openflow_agent          Activated          ofa-0.1.0_46-n3000-SSA-k9.ova
```

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 37.

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 37.

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 37.

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 37.

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	Cisco Nexus 7000 Series Switches Command References

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 3: Feature Information for the Virtual Services Container

Feature Name	Releases	Feature Information
Virtual Services Container	*Cisco Nexus 7000 Series NX-OS*	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.