Cisco Plug-in for OpenFlow Configuration Guide 1.3

First Published: 2014-02-04

Last Modified: 2017-02-17

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

Preface

This preface includes the following sections:

Audience

This guide is intended primarily for data center administrators with responsibilities and expertise in one or more of the following:

- Virtual machine installation and administration
- Server administration
- Switch and network administration

Document Conventions

Command descriptions use the following conventions:

Convention	Description
bold	Bold text indicates the commands and keywords that you enter literally as shown.
Italic	Italic text indicates arguments for which the user supplies the values.
[x]	Square brackets enclose an optional element (keyword or argument).

Convention	Description
[x | y]	Square brackets enclosing keywords or arguments separated by a vertical bar indicate an optional choice.
{x | y}	Braces enclosing keywords or arguments separated by a vertical bar indicate a required choice.
[x {y | z}]	Nested set of square brackets or braces indicate optional or required choices within optional or required elements. Braces and a vertical bar within square brackets indicate a required choice within an optional element.
`variable`	Indicates a variable for which you supply values, in context where italics cannot be used.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.

Examples use the following conventions:

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

This document uses the following conventions:

Note Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the manual.

Caution Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Warning IMPORTANT SAFETY INSTRUCTIONS

This warning symbol means danger. You are in a situation that could cause bodily injury. Before you work on any equipment, be aware of the hazards involved with electrical circuitry and be familiar with standard practices for preventing accidents. Use the statement number provided at the end of each warning to locate its translation in the translated safety warnings that accompanied this device.

SAVE THESE INSTRUCTIONS

Related Documentation for Cisco Nexus 9000 Series Switches

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

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

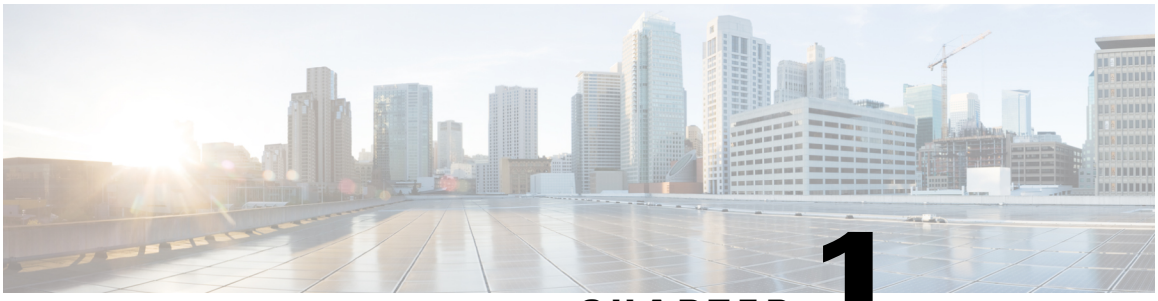
Documentation Feedback

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

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, using the Cisco Bug Search Tool (BST), submitting a service request, and gathering additional information, see *What's New in Cisco Product Documentation* at: http://www.cisco.com/c/en/us/td/docs/general/whatsnew/whatsnew.html

Subscribe to *What's New in Cisco Product Documentation*, which lists all new and revised Cisco technical documentation as an RSS feed and delivers content directly to your desktop using a reader application. The RSS feeds are a free service.



CHAPTER 1

Cisco Plug-in for OpenFlow

This chapter contains the following sections:

Cisco Plug-in for OpenFlow

Cisco Plug-in for OpenFlow, Release 2.1.5 provides better control over networks making them more open, programmable, and application-aware and supports the following specifications defined by the Open Networking Foundation (ONF) standards organization:

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

Prerequisites for Cisco Plug-in for OpenFlow

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

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

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

Table 1: Controller Support

OpenFlow Version	Supported Controllers
OpenFlow 1.0	Cisco Nexus Data Broker (NDB), POX, or Ixia controllers.
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 is 360 MB.

Restrictions for Cisco Plug-in for OpenFlow

- OpenFlow is supported on the following platforms:
  - Cisco Nexus 9300 Series switches
  - Cisco Nexus 3000 Series switches
  - Cisco Nexus 31128PQ switch
  - Cisco Nexus 3232C switch
  - Cisco Nexus 3264Q switch
- OpenFlow is not supported on the following platforms:
  - Cisco Nexus 9500 Series switches
  - Cisco Nexus 3164Q switch
  - Cisco Nexus 9200 Series switches
- Cisco Nexus 3232C and Cisco Nexus 3264Q switches have maximum of 1000 L3 flows in openflow mode and 2000 L3 flows in openflow-lite mode.
- Cisco Nexus 9000 and Cisco Nexus 3000 platforms support OpenFlow in pure Layer 2 forwarding.
- All ports designated for OpenFlow switch have to be Layer 2 physical interfaces or port channels. It needs the interfaces to be configured as a trunk port.
- Layer 3 interfaces or SVI interfaces are not allowed to be configured as OpenFlow ports ('of-ports').
- OpenFlow hybrid model is supported. VLANs configured for OpenFlow logical switch should not overlap with regular switch interfaces.
- PACL on a Layer 2 port that is also configured to be an OF port is not supported.
- RAGUARD on a Layer 2 port that is also configured to be an OF port is not supported.
- Fabric Extenders (FEXs) are not supported.
- Port channels consisting of ports in the following modules are not supported:
  - N9K-M12PQ

  - N9K-C9372PX 40g ports
  - N9K-C9372TX 40g ports
  - N9K-C9332PQ ports 13-14, 27-32

- N3K-C3164Q-40GE is not supported.
- Cisco Plug-in for OpenFlow supports only a subset of OpenFlow 1.3 and OpenFlow 1.0 functions. For more information, see Cisco Plug-in for OpenFlow Feature Support, on page 4.
- You cannot configure more than one Cisco Plug-in for OpenFlow logical switch. The logical switch ID has a value of 1.
- OpenFlow hybrid model (ships-in-the-night) is supported. VLANs configured for Cisco Plug-in for OpenFlow logical switch ports should not overlap with regular device interfaces.
- Cisco Plug-in for OpenFlow logical switch ports must not be configured in a mode other than trunk port.
- You cannot configure a bridge domain, Virtual LANs and virtual routing and forwarding (VRF) interfaces on an Cisco Plug-in for OpenFlow logical switch. You can configure only Layer 2 physical interfaces or port-channel interfaces.
- You cannot configure more than 512 VLANs in Per-VLAN Spanning Tree+ (PVST+) mode.
- Matching of flows that use IPv6 address fields and ports is not supported. Connection to controller using IPv6 addresses is not supported. IPv6 Ethertype is supported.
- Cisco IOS In-Service Software Upgrade (ISSU) is not supported for Cisco Plug-in for OpenFlow.
- MIBs and XMLs are not supported
- You cannot configure more than 1400 MAC flows in the ACL table for Cisco Nexus 3000 Series switches. However, you cannot configure more than 700 ACL flows for Cisco Nexus 3000 Series switches with double-wide TCAM carving configuration for a 12-tuple match.

  For Cisco Nexus 3172, you can configure a maximum of 3000 ACL flows normally and a maximum of 1500 ACL flows with double-wide TCAM configuration. For Cisco Nexus 3548, you can configure a maximum of 4095 ACL FIB flows.
- You cannot configure more than 32,000 flows in the MAC forwarding table for the Cisco Nexus 9000 Series switches.
- For Cisco Nexus 3000 Series platforms, MAC forwarding table scale is verified up to 16,000 flows.
- TCAM carving must be non-zero for the QoS region to ensure that control plane policing for selfIp is effective on the Cisco Nexus 3000 Series switches.
- Reachability to controller via Switched Virtual Interface (SVI) is not supported.
- You must not add or remove an interface as a port of a Cisco Plug-in for OpenFlow if the Cisco Plug-in for OpenFlow is inactive or not running.
- You cannot connect to OpenFlow 1.0 and OpenFlow 1.3 controllers simultaneously. All controllers must support the same version.
- The minimum idle timeout for flows must be 120 seconds.
- LACP port-channels are not supported for OpenFlow. Remove all OpenFlow related configurations and uninstall the OVA virtual service before downgrading to an earlier release.

Information About Cisco Plug-in for OpenFlow

Cisco Plug-in for OpenFlow Feature Support

The following is a subset of OpenFlow 1.3 and OpenFlow 1.0 functions that are supported by Cisco Plug-in for OpenFlow.

Supported Feature	Additional Notes
The OpenFlow hybrid (ships-in-night) model is supported using the OpenFlow packet format	OpenFlow-hybrid models where traffic can flow between Cisco Plug-in for OpenFlow ports and regular interfaces (integrated) are not supported. Both types of ports can transmit and receive packets. **Note** VLANs must be configured such that the VLANs on the Cisco Plug-in for OpenFlow do not overlap with those on the regular device interfaces.
Configuration of port-channel and physical interfaces as Cisco Plug-in for OpenFlow logical switch ports	• Bridge domain, Virtual LANs and Virtual Routing and Forwarding (VRF) interfaces are not supported. • Only L2 interfaces can be Cisco Plug-in for OpenFlow Logical switch ports.
Configuration of VLANs for each port of the Cisco Plug-in for OpenFlow logical switch	Total number of VLANs across all ports cannot exceed 32000. Maximum VLAN range supported is 4000. You can configure 8 such ports on the Cisco Plug-in for OpenFlow device. Recommended VLAN range supported is 512. You can configure 62 such ports on the Cisco Plug-in for OpenFlow device. VLAN range greater than 512 is not supported in Per-VLAN Spanning Tree+ (PVST+) mode.

Supported Feature	Additional Notes
Pipelines for Cisco Plug-in for OpenFlow Logical Switch	• Pipelines are mandatory for the logical switch. • The logical switch supports two pipelines: one with an L3 ACL forwarding Table and one with both an L3 ACL forwarding table and L2 MAC forwarding table. ◦ Pipeline 201 supports the L3 ACL forwarding table. ◦ Pipeline 202 supports an L3 ACL forwarding table and an L2 MAC forwarding table. Mandatory matches and actions in both tables must be specified in all configured flows. ◦ Pipeline 203, which is supported only on the Nexus 3500 Series switches, supports an L3 ACL forwarding table.

Supported Feature	Additional Notes
L3 ACL Forwarding Table (Match Criteria)	The following match criteria are supported: • Ethertype **Note** For Cisco Nexus 3000 Series switches, you can now use the Ethertype field as a wildcard match criteria when the size of the TCAM is configured for double wide interface ACLs. • Ethernet MAC destination (Supported on Nexus 3000 and 3500 Series switches only) **Note** To keep the field set unique in each table in Pipeline 202, match on destination MAC address is not supported in the ACL table when using Pipeline 202 for Cisco Nexus 3000. • Ethernet MAC source (Supported on Nexus 3000 and 3500 Series switches only) **Note** Cisco Nexus 3000 Series switches support OpenFlow 12-tuple match. To accommodate the additional match criteria of source and destination MAC addresses, the Nexus 3000 switch supports a new TCAM region, **ifacl double-wide**, which is a double-wide interface ACL. • VLAN ID (for IPv4 packets only) • VLAN priority (Supported for the Ethertype value 0x0800 (IP) only) **Note** Not supported on Cisco Nexus 3548 and 3548-X. • IPv4 source address (Supported for the Ethertype value 0x0800 (IP) only) • IPv4 destination address (Supported for the Ethertype value 0x0800 (IP) only) • IP DSCP (Supported for the Ethertype value 0x0800 (IP) only) • IP protocol (Supported for the Ethertype value 0x0800 (IP) only) • Layer 4 source port (Supported for the Ethertype value 0x0800 (IP) only) • Layer 4 destination port (Supported for the Ethertype value 0x0800 (IP) only)

Supported Feature	Additional Notes
L3 ACL Forwarding Table (Action Criteria)	The following action criteria are supported: • Output to single port • Output to a specified interface • Output to controller (OpenFlow Packet-In message) • Rewrite source MAC address (SMAC) ◦ Not supported on the Nexus 5000 series ◦ Supported for the Ethertype value 0x0800 (IP) only • Rewrite destination MAC address (DMAC) ◦ Not supported on the Nexus 5000 series ◦ Supported for the Ethertype value 0x0800 (IP) only • Rewrite VLAN ID ◦ Not supported on the Nexus 5000 series ◦ Supported for the Ethertype value 0x0800 (IP) only • Strip VLAN (Supported for the Ethertype value 0x0800 (IP) only) **Note** Support for strip VLAN on the Cisco Nexus 3548 begins with NX-OS software release 6.0(2)A6(3). • Drop **Note** Rewrite DMAC and Rewrite SMAC actions must be specified together.
L2 MAC Forwarding Table	Match Criteria: • Destination MAC address (mandatory) • VLAN ID (mandatory) Action Criteria: • Output to one port • Drop • Punt-to-controller

Supported Feature	Additional Notes
Default Forwarding Rule	All packets that cannot be matched to flows are dropped by default. You can configure sending unmatched packets to the controller.
OpenFlow 1.3 message types	The “modify state” and “queue config” message types are not supported. All other message types are supported.
Connection to up to eight controllers	Transport Layer Security (TLS) is supported for the connection to the controller.
Multiple actions	If multiple actions are associated with a flow, they are processed in the order specified. The output action should be the last action in the action list. Any action after the output action is not supported, and can cause the flow to fail and return an error to the controller. Flows defined on the controller must follow the following guidelines : • The flow can have only up to 16 output actions. • The flow should have the output action at the end of all actions. • The flow should not have multiple rewrite actions that override one another. For example, strip VLAN after set VLAN or multiple set VLANs. **Note** Support for strip VLAN and set VLAN on the Cisco Nexus 3548 begins with NX-OS software release 6.0(2)A6(3). • The flow should not have an output–to–controller action in combination with other output–to–port actions or with VLAN–rewrite actions. • Flows with unsupported actions will be rejected.
Supported counters	Per Table—Active Entries, Packet Lookups, Packet Matches. Per Flow—Received Packets. Per Port—Received or Transmitted packets, bytes, drops and errors.

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.

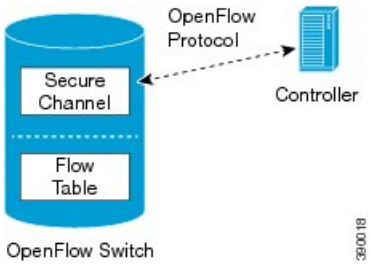
Cisco Plug-in for OpenFlow 1.1.5 refers to Cisco Plug-in for OpenFlow, Release 1.1.5.

A controller can be Cisco Nexus Data Broker (NDB), 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, modification, or removal of flows, and the handling of OpenFlow 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.

OFA Decommissioning

OFA must be un-configured before the virtual service is de-activated and uninstalled. If this is not done, part of the OpenFlow configuration on the interfaces will persist even after decommissioning OFA.

How to Configure Cisco Plug-in for OpenFlow

Configuring Physical Device Parameters

Adjusting the Number of Flow Entries (Nexus 3000 Series and Nexus 3100 Series)

You can use this task to adjust the number of L3 flow entries. By default, 384 flow entries are supported. You can adjust the number of flow entries in a Nexus 3000 Series device to the maximum (1400), using the steps listed below. You can use similar steps to adjust the number of flow entries in a Nexus 3100 Series device to the maximum (3000).

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**hardware profile tcam region vacl 0** **Example:** `Device(config)# hardware profile tcam region vacl 0`	Configures the size of TCAM region for VLAN Access Control Lists (ACLs).
Step 4	**hardware profile tcam region e-vacl 0** **Example:** `Device(config)# hardware profile tcam region e-vacl 0`	Configures the size of TCAM region for egress VLAN ACLs.
Step 5	**hardware profile tcam region racl 0** **Example:** `Device(config)# hardware profile tcam region racl 0`	Configures the size of TCAM region for router ACLs.

	Command or Action	Purpose
Step 6	**hardware profile tcam region e-racl 0** **Example:** `Device(config)# hardware profile tcam region e-racl 0`	Configures the size of TCAM region for egress router ACLs.
Step 7	**hardware profile tcam region qos 256** **Example:** `Device(config)# hardware profile tcam region qos 256`	Configures the size of TCAM region for QoS.

Step 8	Enter one of the following commands:
	• **hardware access-list tcam region openflow 1408**
	• **hardware access-list tcam region openflow 1408 double-wide**
	
	Example:
	`Device(config)# hardware access-list tcam region openflow 1408`
	
	Example:
	`Device(config)# hardware access-list tcam region openflow 1408 double-wide`
		Configures the size of TCAM region for interface ACLs.
		To accommodate the additional match criteria of source and destination MAC addresses, the Cisco Nexus 3000 switch supports a new TCAM region, **ifacl double-wide**, which is a double-wide interface ACL.
		The **ifacl** and **ifacl double-wide** sizes for Cisco Nexus 3172 are 3072 and 1536, respectively.
		Note To activate the TCAM regions, a reload is needed for the Cisco Nexus 9000 Series.

Step 9	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and enters privileged EXEC mode.
Step 10	**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.
Step 11	**reload** **Example:** `Device# reload`	Reloads the operating system of a device so that virtual-services container support for the device hardware can start.

What to Do Next

Configure global variables for Cisco Plug-in for OpenFlow logical switch.

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.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**no cdp enable** **Example:** `Device(config)# no cdp enable`	Disables Cisco Discovery Protocol (CDP).
Step 3	**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 4	**end** **Example:** `Device(config-vlan)# exit`	Exits VLAN configuration mode and enters privileged EXEC mode.
Step 5	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves the change persistently 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 13.
- Management Interface. Refer to Specifying a Route to a Controller Using a Management Interface, on page 14.

The IP address of the controller is configured in the Configuring a Cisco Plug-in for OpenFlow Logical Switch , on page 18 section.

Specifying a Route to a Controller Using a Physical Interface

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface Ethernet1/1`	Enters the physical interface. The interface used here should not be an 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.
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 through reboots and restarts 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

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *management-interface-name number* **Example:** `Device(config)# interface mgmt0`	Enters the management interface.
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.
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.

	Command or Action	Purpose
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

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 or port-channel 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 18 section.

Configuring a Physical Interface in Layer 2 mode

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

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface Ethernet5/23`	Specifies the interface for the logical switch and enters interface configuration mode.
Step 4	**channel-group** *group-number* **Example:** `Device(config-if)# channel-group 2`	(Optional) Adds the interface to a port-channel.

	Command or Action	Purpose
Step 5	**switchport** **Example:** `Device(config-if)# switchport`	Specifies an interface as a Layer 2 port.
Step 6	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Specifies an interface as a trunk port. • A trunk port can carry traffic of one or more VLANs on the same physical link. (VLANs are based on the trunk-allowed VLANs list.) By default, a trunk interface carries traffic for all VLANs.
Step 7	**switchport mode trunk allowed vlan** [*vlan-list*] **Example:** `Device(config-if)# switchport trunk allowed vlan 1-3`	Sets the list of allowed VLANs that transmit traffic from this interface in tagged format when in trunking mode.
Step 8	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Enables the interface.
Step 9	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.
Step 10	**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

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 Port-Channel Interface

Perform the task below to create a port-channel interface for a Cisco Plug-in for OpenFlow logical switch.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *number* **Example:** `Device(config)# interface port-channel 2`	Specifies the interface for the logical switch and enters interface configuration mode.
Step 4	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Specifies the interface as an Ethernet trunk port. A trunk port can carry traffic in one or more VLANs on the same physical link (VLANs are based on the trunk-allowed VLANs list). By default, a trunk interface can carry traffic for all VLANs. **Note** If the port-channel is specified as a trunk interface, ensure that member interfaces are also configured as trunk interfaces.
Step 5	**switchport mode trunk allowed vlan** [*vlan-list*] **Example:** `Device(config-if)# switchport trunk allowed vlan 1-3`	Sets the list of allowed VLANs that transmit traffic from this interface in tagged format when in trunking mode.
Step 6	**end** **Example:** `Device(config-if)# end`	Ends 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 through reboots and restarts by copying the running configuration to the startup configuration.

What to Do Next

Install and activate Cisco Plug-in for OpenFlow.

Installing and Activating Cisco Plug-in for OpenFlow

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

Before installing and activating Cisco Plug-in for OpenFlow, ensure that an OVA package compatible with the device exists on a connected FTP server. Refer to the Prerequisites for a Virtual Services Container, on page 35. 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 36, 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 38, 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 39, 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.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**pipeline** *pipeline-id* **Example:** `Device(config-ofa-switch)# pipeline 201`	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.

	Command or Action	Purpose

Step 4	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.
		• An interface configured for a port channel should not be configured as an Cisco Plug-in for OpenFlow logical switch port.
		• Repeat this step to configure additional interfaces.

Step 5	**protocol-version** *version-info* **Example:** `Device(config-openflow-switch)# protocol-version 1.0`	Configures the protocol version. • Supported values are: ◦ **1.0—**Configures device to connect to 1.0 controllers only ◦ **1.3—**Configures device to connect to 1.3 controllers only ◦ **negotiate—**Negotiates the protocol version with the controller. Device uses 1.3 for negotiation. **Note** The default value is **negotiate**. • **drop** is the default action for both tables or pipeline 1. This can be overridden by this configuration or the controller.
Step 6	**controller ipv4** *ip-address* [**port** *tcp-port*] [**vrf** *vrf-name*] **security**{**none** | **tls**}	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.

	Command or Action	Purpose
	Example: Controller in default VRF: `Device(config-openflow-switch)# controller ipv4 10.1.1.2 security none`	• 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 7	**default-miss cascade** { **drop** | **controller** | **normal** | } **Example:** `Device(config-ofa-switch)# default-miss cascade 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. • Configuring this step with the **normal** keyword is necessary for pipeline 202 (ACL Table) to add a default permit rule instead of the default drop rule.
Step 8	**tls trust-point local** *local-trust-point* **remote** *remote-trust-point* **Example:** `Device(config-ofa-switch)# tls trust-point local mylocal remote myremote`	(Optional) Specifies the local and remote TLS trustpoints to be used for the controller connection. • For information on configuring trustpoints, refer to PKI Trustpool Management in the PKI Configuration guide.
Step 9	**logging flow-mod** **Example:** `Device(config-ofa-switch)# logging flow-mod`	(Optional) Enables logging of flow changes, including addition, deletion, and modification of flows. • Logging of flow changes is disabled by default. • Flow changes are logged in syslog and can be viewed using the **show logging** command. • Logging of flow changes is a CPU intensive activity and should not be enabled for networks greater than 1000 flows.
Step 10	**probe-interval** *probe-interval* **Example:** `Device(config-openflow-switch)# probe-interval 5`	(Optional) Configures the interval, in seconds, at which the controller is probed. • The default value is 5.

	Command or Action	Purpose
		• The range is from 5 to 65535.
Step 11	**rate-limit packet_in** *controller-packet-rate* **burst** *maximum-packets-to-controller* **Example:** `Device(config-openflow-switch)# rate-limit packet_in 1 burst 4`	(Optional) Configures the maximum packet rate of the connection to the controller and the maximum packets permitted in a burst of packets sent to the controller in a second. • The default value is zero, meaning that an indefinite packet rate and packet burst are permitted. • This rate limit is for Cisco Plug-in for OpenFlow. It is not related to the rate limit of the device (data plane) configured by COPP.
Step 12	**max-backoff** *backoff-timer* **Example:** `Device(config-openflow-switch)# max-backoff 8`	(Optional) Configures the time, in seconds, for which the device must wait before attempting to initiate a connection with the controller. • The default value is eight. • The range is from 1 to 65535.
Step 13	**datapath-id** *id* **Example:** `Device(config-openflow-switch)# datapath-id 111`	(Optional) *id* is a 64bit hex value. A valid *id* is in the range [0x1-0xffffffffffffffff]. This identifier allows the controller to uniquely identify the device.
Step 14	**protocol-version** [**1.0** | **1.3** | **negotiate**] **Example:** `Device(config-openflow-switch)# protocol-version 1.0`	(Optional) This command forces a specific version of the controller connection. If you force version 1.3 and the controller supports only 1.0, no session is established (or vice versa). The default behavior is to negotiate a compatible version between the controller and device.
Step 15	**shutdown** **Example:** `Device(config-openflow-switch)# shutdown`	(Optional) This disables the OpenFlow switch without having to remove all the other configuration.
Step 16	**statistics collection-interval** *seconds* **Example:** `Device(config-openflow-switch)# statistics collection 10`	(Optional) A setting of zero disables statistics collection. This number can be used to reduce the CPU load from periodic stats polling. For example, if you have 1000 flows and choose a stats collection interval of 10 seconds, 1000flows/10s = 100 flows per second poll rate.

	Command or Action	Purpose
		Note Each flow table has a prescribed maximum flows-per-second poll rate supported by hardware as displayed in the **show openflow hardware capabilities** command. If you choose a stats collection interval that is too small, the maximum rate supported by the hardware is used, effectively throttling the stats collection.
Step 17	**default-miss** *value* **Example:** `Device(config-openflow-switch)# default-miss continue-normal`	(Optional) The **default-miss** command sets the behavior when a packet does not match a flow in the flow table. **Note** Not every action is supported on every platform. **continue-drop**: a miss in a flow table will cascade to perform a match in the next table (if applicable). A miss in the terminal table in the pipeline will result in the packet being dropped. **continue-normal**: a miss in a flow table will cascade to perform a match in the next table (if applicable). A miss in the terminal table in the pipeline will result in the packet being sent to the switch's normal hardware processing. **continue-controller**: a miss in a flow table will cascade to perform a match in the next table (if applicable). A miss in the terminal table in the pipeline will result in the packet being sent to the controller. Configuring this sets the behavior when a packet does not match a flow in the flow table. **drop**: a miss in the first flow table of the pipeline will not cascade to any other table. Instead the packet will be dropped. **normal**: a miss in the first flow table of the pipeline will not cascade to any other table. Instead the packet will be sent to the switch's normal hardware forwarding. **controller**: a miss in the first flow table of the pipeline will not cascade to any other table. Instead the packet will be sent to the controller.
Step 18	**end** **Example:** `Device(config-openflow-switch)# end`	Exits logical switch configuration mode and enters privileged EXEC mode.
Step 19	**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

Verify Cisco Plug-in for OpenFlow.

Verifying Cisco Plug-in for OpenFlow

Procedure

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: 201
  Signal version: Openflow 1.0
  Data plane: secure
  Table-Miss default: NONE
  Config state: no-shutdown
  Working state: enabled
  Rate limit (packet per second): 0
  Burst limit: 0
  Max backoff (sec): 8
  Probe interval (sec): 5
  TLS local trustpoint name: not configured
  TLS remote trustpoint name: not configured
  Stats coll. period (sec): 5
  Logging flow changes: Disabled
  OFA Description:
    Manufacturer: Cisco Systems, Inc.
    Hardware: N3K-C3064PQ V01
    Software: 6.0(2)U2(1) of_agent 1.1.0_fc1
    Serial Num: SSI15200QD8
    DP Description: n3k-200-141-3:sw1
  OF Features:
    DPID:0001547fee00c2a0
    Number of tables:1
    Number of buffers:256
    Capabilities: FLOW_STATS TABLE_STATS PORT_STATS
```

```
    Actions: OUTPUT SET_VLAN_VID STRIP_VLAN SET_DL_SRC SET_DL_DST
  Controllers:
    1.1.1.1:6653, Protocol: TLS, VRF: s
  Interfaces:
    Ethernet1/1
    Ethernet1/7
```

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: 1p
  Controller: 1
    10.5.84.254:6633
    Protocol: tcp
    VRF: default
    Connected: No
    Role: Master
    Negotiated Protocol Version: disconnected
    Last Alive Ping: 07/04/2014 06:55:42
    last_error:Connection timed out
    state:CONNECTING
    sec_since_connect:291686
    sec_since_disconnect:8
```

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: 1
  Controller: 1
    address                          :  ssl:10.1.1.1:6653
    connection attempts              :  181
    successful connection attempts   :  0
    flow adds                        :  0
    flow mods                        :  0
    flow deletes                     :  0
    flow removals                    :  0
    flow 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

Device# show openflow switch 1 controllers stats

Logical Switch Id: 1
Total Controllers: 1
  Controller: 1
    address                          :  tcp:10.5.84.254:6633
    connection attempts              :  16927
    successful connection attempts   :  1
    flow adds                        :  1
    flow mods                        :  0
    flow deletes                     :  0
    flow removals                    :  0
    flow errors                      :  1
    flow unencodable errors          :  0
    total errors                     :  2
    echo requests                    :  rx: 2099, tx: 2137
    echo reply                       :  rx: 2136, tx: 2099
    flow stats                       :  rx: 0, tx: 0
```

```
    barrier                          :  rx: 0, tx: 0
    packet-in/packet-out             :  rx: 0, tx: 2099
```

Step 4 **show openflow switch** *switch-id* **ports**

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
   2  Ethernet1/2       PORT_UP        LINK_UP        10MB-FD
   3  Ethernet1/3       PORT_UP        LINK_DOWN      100MB-HD AUTO_NEG
   4  Ethernet1/4       PORT_UP        LINK_UP        10MB-FD
```

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

Displays flows defined for the device by controllers.

Example:

```
Device# show openflow switch 1 flows

Total flows: 2
Flow: 1
  Rule:              ip,dl_vlan=99
  Actions:           strip_svlan,output:1
  Priority:          0x8000
  Table:             0
  Cookie:            0x466c6f7732
  Duration:          96.359s
  Number of packets: 0
  Number of bytes:   0

Flow: 2
  Rule:              ip,in_port=2,dl_vlan=50
  Actions:           output:1
  Priority:          0x8000
  Table:             0
  Cookie:            0x1
  Duration:          95.504s
  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: 1
  Port 31: rx pkts=36688, bytes=7204655, drop=0, errs=0,
           tx pkts=0, bytes=3473880, drop=0, errs=0,
Total tables: 1
  Table 0: classifier
  Wildcards = 0x3fffff
  Max entries =   1500
  Active entries = 0
  Number of lookups = 0
  Number of matches = 0
```

Flow statistics are available for pipeline 201 and table 0. For pipeline 202, flow statistics are not available for table 1.

Step 7 **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
```

Step 8 **show running-config | section openflow**
Displays configurations made for Cisco Plug-in for OpenFlow.

Example:

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

  openflow
    switch 1
      pipeline 201
      controller ipv4 10.86.201.162 port 8050 vrf management security none
      of-port interface ethernet1/1
      of-port interface ethernet1/2
      of-port interface ethernet1/3
      of-port interface ethernet1/37
      of-port interface ethernet1/4
```

Step 9 **show openflow hardware capabilities**

Displays Cisco Plug-in for OpenFlow configurations.

Example:

```
Device# show openflow hardware capabilities
  Pipeline ID: 201

    Flow table ID: 0

    Match Capabilities                  Match Types
    ------------------                  -----------
    ethernet type                       mandatory
    VLAN ID                             optional
    VLAN priority code point            optional
    IP DSCP                             optional
    IP protocol                         optional
    IPv4 source address                 lengthmask
    IPv4 destination address            lengthmask
    source port                         optional
    destination port                    optional
    in port (virtual or physical)       optional

    Actions:
        output to: specified interface, use normal forwading, controller
        set: set eth source mac, set eth destination mac, set vlan id
        pop: pop vlan tag
        other actions: drop packet



  Pipeline ID: 202

    Flow table ID: 0

    Match Capabilities                  Match Types
    ------------------                  -----------
    ethernet type                       mandatory
    VLAN ID                             optional
    VLAN priority code point            optional
    IP DSCP                             optional
    IP protocol                         optional
    IPv4 source address                 lengthmask
    IPv4 destination address            lengthmask
    source port                         optional
    destination port                    optional
    in port (virtual or physical)       optional

    Actions:
        output to: specified interface, use normal forwading, controller
        set: set eth source mac, set eth destination mac, set vlan id
        pop: pop vlan tag
        other actions: drop packet
```

```
Flow table ID: 1

Match Capabilities                  Match Types
------------------                  -----------
ethernet mac destination            mandatory
VLAN ID                             mandatory

Actions:
    output to: specified interface
    other actions: drop packet
```

Configuration Examples for Cisco Plug-in for OpenFlow

Example: Enabling Hardware Support for Cisco Plug-in for OpenFlow

```
Device> enable
Device# configure terminal
! Enables support for OpenFlow VLAN tagging actions.
Device(config)# hardware profile openflow
Device# copy running-config startup-config
Device# reload
```

Example: Adjusting the Number of Flow Entries

```
Device> enable
Device# configure terminal
Device(config)# hardware profile tcam region vacl 0
Device(config)# hardware profile tcam region e-racl 0
Device(config)# hardware profile tcam region e-vacl 0
Device(config)# hardware profile tcam region racl 256
Device(config)# hardware profile tcam region ifacl 1664
Device(config)# exit
Device# copy running-config startup-config
Device# reload
```

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

```
Device# configure terminal
Device(config)# mac-learn disable
Device(config)# spanning-tree mode mst
Device(config)# vlan 2
Device(config-vlan)# end
```

Example: Configuring Control Plane Policing for Packets Sent to a Controller

```
Device# configure terminal
Device# setup


         ---- Basic System Configuration Dialog ----

This setup utility will guide you through the basic configuration of
the system. Setup configures only enough connectivity for management
of the system.

*Note: setup is mainly used for configuring the system initially,
when no configuration is present. So setup always assumes system
defaults and not the current system configuration values.

Press Enter at anytime to skip a dialog. Use ctrl-c at anytime
to skip the remaining dialogs.
```

```
Would you like to enter the basic configuration dialog (yes/no): yes

  Create another login account (yes/no) [n]:

  Configure read-only SNMP community string (yes/no) [n]:

  Configure read-write SNMP community string (yes/no) [n]:

  Enter the switch name : QI32

  Continue with Out-of-band (mgmt0) management configuration? (yes/no) [y]: n

  Configure the default gateway? (yes/no) [y]: n

  Enable the telnet service? (yes/no) [n]: y

  Enable the ssh service? (yes/no) [y]: n

  Configure the ntp server? (yes/no) [n]:

  Configure default interface layer (L3/L2) [L2]:

  Configure default switchport interface state (shut/noshut) [noshut]:
  Configure CoPP System Policy Profile ( default / l2 / l3 ) [default]:

The following configuration will be applied:
  switchname QI32
  telnet server enable
  no ssh server enable
  system default switchport
  no system default switchport shutdown
  policy-map type control-plane copp-system-policy ( default )

Would you like to edit the configuration? (yes/no) [n]:

Use this configuration and save it? (yes/no) [y]:

[########################################] 100%
Copy complete, now saving to disk (please wait)...

Device# configure terminal
Device(config)# policy-map type control-plane copp-system-policy
Device(config-pmap)# class copp-s-dpss
Device(config-pmap-c)# police pps 1000
Device(config-pmap-c)# end
Device# show run copp
```

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

```
Device# configure terminal
Device(config)# interface Ethernet1/1
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# 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)# vrf context management
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: 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# configure terminal

Device(config)# interface ethernet1/1
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# exit

Device(config)# interface ethernet1/2
! Adding the interface to a port channel.
Device(config-if)# channel-group 2
Device(config-if)# switchport mode trunk
Device(config-if)# no shutdown
Device(config-if)# end
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)# end
Device# copy running-config startup-config
```

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

```
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 201
Device(config-ofa-switch)# pipeline 1
Device(config-ofa-switch)# logging flow-mod
Device(config-ofa-switch)# tls trust-point local local-trustpoint-name remote
remote-trustpoint-name
Device(config-ofa-switch)# max-backoff 5
Device(config-ofa-switch)# probe-interval 5
Device(config-ofa-switch)# rate-limit packet-in 30 burst 50
Device(config-ofa-switch)# controller ipv4 10.0.1.6 security none
! Adding an interface to the Cisco Plug-in for OpenFlow logical switch.
Device(config-ofa-switch)# of-port interface ethernet1/1
Device(config-ofa-switch)# of-port interface ethernet1/2

! 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# configure terminal
Device(config)# openflow
Device(config-ofa)# switch 1
Device(config-ofa-switch)# pipeline 201
! Specifying a controller that is part of a VRF.
```

```
Device(config-ofa-switch)# controller ipv4 10.0.1.6 vrf mgmtVrf security none
! Adding an interface to the Cisco Plug-in for OpenFlow logical switch.

Device(config-ofa-switch)# of-port interface ethernet1/1
Device(config-ofa-switch)# of-port interface ethernet1/2


! 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
```

Additional Information for Cisco Plug-in for OpenFlow

Related Documents

Related Topic	Document Title
Cisco commands	Cisco Nexus 3000 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

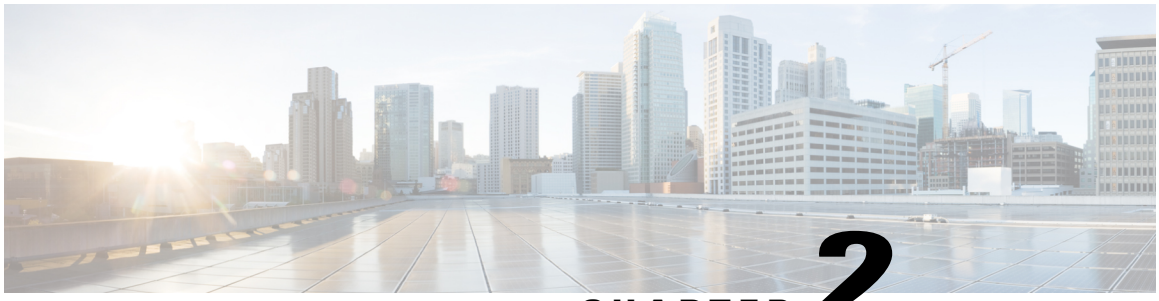
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 2: Feature Information for Cisco Plug-in for OpenFlow

Releases	Supported Platforms	Feature Information
Cisco Plug-in for OpenFlow Release 1.3	The supported platforms • Nexus 3000 Series Devices • Nexus 3100 Series Devices • Nexus 9300 Series Devices	For Cisco Nexus 3000 and Cisco Nexus 3100 Series devices, the Cisco Plug-in for OpenFlow Release 1.3 needs to be used for NX-OS release 7.0(3) and later.
Cisco Plug-in for OpenFlow Release 1.1.5	The supported platforms are Nexus 3000 Series Devices. The Nexus 3548-X device is supported in NX-OS software release 6.0(2)A6(2) and higher.	Cisco Plug-in for OpenFlow supports OFA decommissioning.
Cisco Plug-in for OpenFlow Release 1.1.1	The supported platforms are: • Nexus 3000 Series Devices • Nexus 5000 Series Devices • Nexus 6000 Series Devices	Cisco Plug-in for OpenFlow now supports Nexus 5000 and 6000 Series.

Releases	Supported Platforms	Feature Information
Cisco Plug-in for OpenFlow Release 1.1	The supported platforms are Nexus 3000 Series Devices.	• The OpenFlow hybrid (ships-in-night) model is supported. • L3 ACL and L2 MAC forwarding tables are supported and can be configured using pipelines. • Transport Layer Security (TLS) is supported in Cisco Plug-in for OpenFlow and controller communications. • VLAN priority has been introduced as a flow action. The following commands have been introduced: **clear openflow, max-backoff, probe-interval, rate-limit, tls trust-point**. The **controller** command has been modified to include the **no-tls** keyword.
Cisco Plug-in for OpenFlow Release 1.0.1	The supported platforms are Nexus 3000 Series Devices.	The following flow actions are supported: • Modify source MAC address • Modify destination MAC address
Cisco Plug-in for OpenFlow Release 1.0	The supported platforms are Nexus 3000 Series Devices.	Cisco Plug-in for OpenFlow supports OpenFlow 1.0, and helps networks become more open, programmable, and application-aware.



CHAPTER 2

Virtual Services Container

This chapter contains the following sections:

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 36 (required)
- Deactivating and Uninstalling an Application from a Virtual Services Container, on page 38
- Upgrading an Application in a Virtual Services Container, on page 39
- Collecting General Troubleshooting Information, on page 41
- Verifying Virtual Services Container Applications , on page 43

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.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**copy** *from*:*//source-directory-url destination-directory-url* **Example:** `Device# copy tftp://myserver.com/downloads/ofa-1.0.0-n3000-SPA-k9.ova bootflash:/ofa-1.0.0-n3000-SPA-k9.ova`	Downloads the new OVA package to the device for upgrade. Possible values are: • sftp: • tftp:

	Command or Action	Purpose
		• ftp: • http: • bootflash:
Step 3	**virtual-service install name** *virtual-services-name* **package** *file* **Example:** `Device# virtual-service install name openflow_agent package bootflash:/ofa-1.0.0-n3000-SPA-k9.ova`	Installs an OVA package from the specified location onto a device. Ensure that the ova file is located in the root directory of the storage device • The *virtual-services-name* defined here should be used in all occurrences of this argument in this document.
Step 4	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 5	**virtual-service** *virtual-services-name* **Example:** `Device(config)# virtual-service openflow_agent`	Configures a virtual services container and enters virtual services configuration mode. • Use the *virtual-services-name* defined during installation of the application. • Ensure that installation is complete before proceeding to the next step using the **show virtual-service list** command.
Step 6	**activate** **Example:** `Device(config-virt-serv)# activate`	Activates the installed virtual services container.
Step 7	**end** **Example:** `Device(config-virt-serv)# end`	Exits virtual services configuration mode and enters privileged EXEC mode.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves the change persistently through reboots and restarts by copying the running

	Command or Action	Purpose
		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.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**virtual-service** *virtual-services-name* **Example:** `Device(config)# virtual-service openflow_agent`	Enters virtual services configuration mode to configure a specified application. • Use the *virtual-services-name* defined during installation of the application.
Step 4	**no activate** **Example:** `Device(config-virt-serv)# no activate`	Disables the application.
Step 5	**no virtual-service** *virtual-services-name* **Example:** `Device(config)# no virtual-service openflow_agent`	Unprovisions the application. • Use the *virtual-services-name* defined during installation of the application. • This command is optional for all devices running Cisco IOS-XE.

	Command or Action	Purpose
Step 6	**end** **Example:** `Device(config-virt-serv)# end`	Exits virtual services configuration mode and enters privileged EXEC mode.
Step 7	**virtual-service uninstall name** *virtual-services-name* **Example:** `Device# virtual-service uninstall name openflow_agent`	Uninstalls the application. • Use the *virtual-services-name* defined during installation of the application. • Run this command only after receiving a successful deactivation response from the device.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Upgrading an Application in a Virtual Services Container

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

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

Procedure

	Command or Action	Purpose
Step 1	**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: • scp: • sftp: • tftp: • ftp: • http: • bootflash:

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**virtual-service** *virtual-services-name* **Example:** `Device(config)# virtual-service openflow_agent`	Enters virtual services configuration mode for configuring a specified application. • Use the *virtual-services-name* defined during installation of the application.
Step 4	**no activate** **Example:** `Device(config-virt-serv)# no activate`	Disables the application.
Step 5	**end** **Example:** `Device(config-virt-serv)# end`	Exits virtual services configuration mode and enters privileged EXEC mode.
Step 6	**virtual-service upgrade name** *virtual-services-name* **package** *file* **Example:** `Device# virtual-service upgrade name openflow_agent package bootflash:/ofa-1.0.0-n3000-SPA-k9.ova`	Upgrades the application using the specified OVA file. • Use the *virtual-services-name* defined during installation of the application. • Run this command only after receiving a successful deactivation message from the device.
Step 7	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 8	**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*

	Command or Action	Purpose
		defined during installation of the application.
Step 9	**activate** **Example:** `Device(config-virt-serv)# activate`	Activates the application.
Step 10	**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.

Procedure

	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`	

	Command or Action	Purpose
	(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	
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 activa ted virtual service 'openflow_agent' 2013 May 8 18:57:15 n3k-202-194-2 %VMAN-5-VIRT_INST: LOG FROM VIRTUAL SERVICE n 3k: 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 n 3k: OVS: sw1<->tcp:10.44.94.173:6633%management:	This command shows the VMAN logging configuration and contents of log files.

	Command or Action	Purpose
	connected 2013 May 9 15:00:05 n3k-202-194-2 %VMAN-5-VIRT_INST: LOG FROM VIRTUAL SERVICE n 3k: OVS: sw1<->tcp:10.168.1.31:7777: connected	
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 Total Entries Written: 2513 One shot mode: No One shot and full: No Disabled: False	This command displays trace settings of a trace buffer.
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

Procedure

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

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 41.

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 41.

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 41.

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 41.

Configuration Examples for a Virtual Services Container

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

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

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

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

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

Additional References for the Virtual Services Container

Related Documents

Related Topic	Document Title
Cisco commands	

Technical Assistance

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

Feature Information for Virtual Services Container

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

Table 3: Feature Information for the Virtual Services Container

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

Glossary

application

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

container

This is another name for virtual service container.

guest

Application instance running within a container.

host

Operating system installed on a device.

KVM

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

LxC

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

logical Switch

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

OVA

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

physical Switch

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

virtual machine

This is another name for virtual service container.

virtual service

This is another name for virtual service container.

virtual services container

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

VMAN

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