# Cisco Nexus 3000 Series NX-OS Interfaces Configuration Guide, Release 6.x

**First Published:** 2013-04-09

**Last Modified:** 2015-03-31

**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

The preface contains the following sections:



## Audience

This publication is for network administrators who configure and maintain Cisco Nexus devices.

## Document Conventions

**Note** As part of our constant endeavor to remodel our documents to meet our customers' requirements, we have modified the manner in which we document configuration tasks. As a result of this, you may find a deviation in the style used to describe these tasks, with the newly included sections of the document following the new format.

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.

# Related Documentation for Cisco Nexus 3000 Series NX-OS Software

The entire Cisco NX-OS 3000 Series documentation set is available at the following URL:

http://www.cisco.com/en/US/products/ps11541/tsd_products_support_series_home.html

### Release Notes

The release notes are available at the following URL:

http://www.cisco.com/en/US/products/ps11541/prod_release_notes_list.html

### Installation and Upgrade Guides

The installation and upgrade guides are available at the following URL:

http://www.cisco.com/en/US/products/ps11541/prod_installation_guides_list.html

### License Information

For information about feature licenses in NX-OS, see the *Cisco NX-OS Licensing Guide*, available at the following URL: http://www.cisco.com/en/US/docs/switches/datacenter/sw/nx-os/licensing/guide/b_Cisco_NX-OS_Licensing_Guide.html.

For the NX-OS end user agreement and copyright information, see *License and Copyright Information for Cisco NX-OS Software*, available at the following URL: http://www.cisco.com/en/US/docs/switches/datacenter/sw/4_0/nx-os/license_agreement/nx-ossw_lisns.html.

### Configuration Guides

The configuration guides are available at the following URL:

http://www.cisco.com/en/US/products/ps11541/products_installation_and_configuration_guides_list.html

### Programming Guides

The XML Interface User Guide and other programming guides are available at the following URL:

http://www.cisco.com/en/US/products/ps11541/products_programming_reference_guides_list.html

### Technical References

The technical references are available at the following URL:

http://www.cisco.com/en/US/products/ps11541/prod_technical_reference_list.html

### Error and System Messages

The error and system message reference guides are available at the following URL:

http://www.cisco.com/en/US/products/ps11541/products_system_message_guides_list.html

# Documentation Feedback

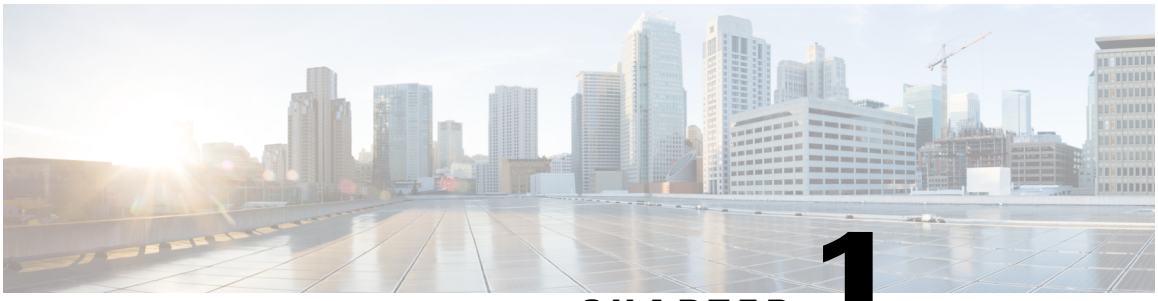
To provide technical feedback on this document, or to report an error or omission, please send your comments to: .

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.

To receive new and revised Cisco technical content directly to your desktop, you can subscribe to the What's New in Cisco Product Documentation RSS feed. RSS feeds are a free service.



CHAPTER 1

# New and Changed Information

This chapter contains the following sections:



# New and Changed Information in this Release

The following table provides an overview of the significant changes made to this configuration guide. The table does not provide an exhaustive list of all changes made to this guide or all new features in a particular release.

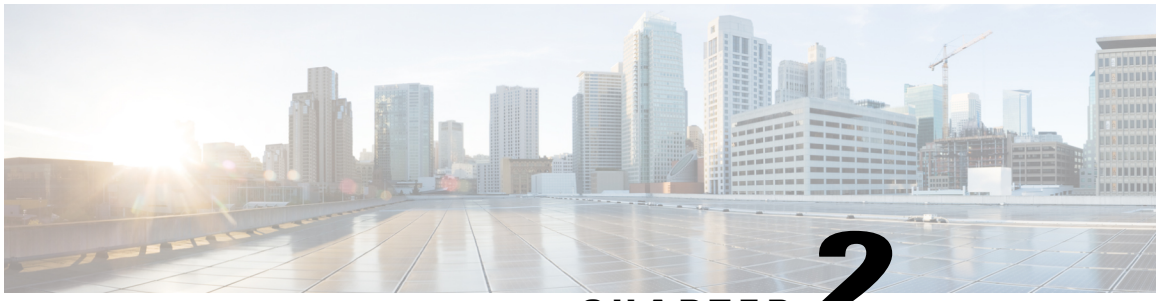
| Feature | Description | Added or Changed in Release | Where Documented |
|---|---|---|---|
| vPC Fast Convergence | A New CLI has been added to enable/disable the vPC optimizations feature. | 6.0(2)U6(4) | Creating a vPC Domain, on page 125 |
| IPv6 over v4 GRE Tunneling | IPv6 in IPv4 with GRE header is now supported. | 6.0(2)U6(1) | Configuring a GRE Tunnel, on page 81 |
| QSFP+ (40-Gb) transceiver | A new QSFP+ (40-Gb) transceiver is now supported on the Cisco Nexus 3000 Series switches. It has to be used in 4 x 10G mode with splitter cable and LR optics. | 6.0(2)U5(1) | 40-Gigabit Ethernet Interface Speed, on page 9 |

| Feature | Description | Added or Changed in Release | Where Documented |
|---|---|---|---|
| Default port mode changed | Starting with Release 6.0(2)U5(1), the default port mode on Cisco Nexus 3132Q and Cisco Nexus 3132CR Series switches after write erase is 32x40G mode. | 6.0(2)U5(1) | Port Modes, on page 10 |
| Global knob for auto-negotiation disable | You can disable auto-negotiation on all 40G interfaces. | 6.0(2)U5(1) | Disabling Link Negotiation, on page 21 |
| MAC-Embedded IPv6 Address | Introduced the MEv6 feature. | 6.0(2)U4(1) | MAC-Embedded IPv6 Address, on page 40 |
| Statistics Collection on Interfaces | Introduced the **load-interval** command to change the sampling interval for statistics collections on interfaces. | 6.0(2)U4(1) | Monitoring Layer 3 Interfaces, on page 51 |
| DHCP Client Configuration | You can now configure the IPv4 or IPv6 address of a DHCP client on a management interface, or a physical Ethernet interface. | 6.0(2)U4(1) | DHCP Client Discovery, on page 39 |
| VXLAN | VXLANs extend Layer 2 networks across the Layer 3 infrastructure using MAC-in-UDP encapsulation and tunneling for Cisco Nexus 3100 Series switches. | 6.0(2)U3(2) | Configuring VXLANs, on page 87 |
| Resilient Hashing | Added support for Cisco Nexus 3172 switches. Resilient hashing maps traffic flows to physical ports and redistributes traffic from failed links uniformly across the working links. | 6.0(2)U3(1) | Resilient Hashing, on page 60 |

| Feature | Description | Added or Changed in Release | Where Documented |
|---|---|---|---|
| Downlink Delay | This feature enables you to operationally enable uplink SFP+ ports before downlink RJ-45 ports after a reload on a Cisco Nexus 3048 switch. | 6.0(2)U3(1) | Downlink Delay, on page 14 |
| DHCP Client Configuration | You can configure the IP address of a DHCP client on SVIs by using the **ip address dhcp** command. | 6.0(2)U3(1) | DHCP Client Discovery, on page 39 |
| Dynamic Port Breakout for Cisco Nexus 3172 | The dynamic breakout feature is now supported by Cisco Nexus 3172. | 6.0(2)U2(3) | Port Modes, on page 10 |
| Symmetric Hashing | Symmetric hashing enables bidirectional traffic to use the same physical interface and maps each physical interface in the port channel to a set of flows. | 6.0(2)U2(3) | Symmetric Hashing, on page 61 |
| Resilient Hashing | Resilient hashing maps traffic flows to physical ports and redistributes traffic from failed links uniformly across the working links. | 6.0(2)U2(1) | Resilient Hashing, on page 60 |
| Hashing for NVGRE Traffic | Hashing for NVGRE traffic allows the switch to include the GRE Key field present in the GRE header in hash computations when NVGRE traffic is forwarded over a port channel or an Equal Cost Multipath (ECMP). | 6.0(2)U2(1) | Hashing for NVGRE Traffic, on page 60 |

| Feature | Description | Added or Changed in Release | Where Documented |
|---|---|---|---|
| Dynamic Port Breakout for Cisco Nexus 3132 | The dynamic port breakout feature enables:<br>• A 40-GbE port to break out into four 10-GbE ports<br>• Four 10-GbE ports to break into a 40-GbE port | 6.0(2)U2(1) | 40-Gigabit Ethernet Interface Speed, on page 9 |
| Consistency Checkers | The following consistency checkers were introduced to check for consistency and display the results:<br>• Port Channel Membership Consistency Checker<br>• Layer 3 Interface Consistency Checker<br>• Link State Consistency Checker | 6.0(2)U2(1) | • Triggering the Port Channel Membership Consistency Checker, on page 73<br>• Triggering the Layer 3 Interface Consistency Checker, on page 51<br>• Triggering the Link State Consistency Checker, on page 16 |
| SVI Autostate Disable | The SVI Autostate Disable feature enables the Switch Virtual Interface (SVI) to be in the “up” state even if no interface is in the “up” state in the corresponding VLAN. | 6.0(2)U2(1) | SVI Autostate Disable, on page 39 |
| IP-in-IP encapsulation and decapsulation tunnel support | The IP-in-IP encapsulation and decapsulation tunnel support allows you to configure the way in which encapsulated packets are sent from and delivered to tunnel interfaces. | 6.0(2)U2(1) | Configuring IP Tunnels, on page 75 |

| Feature | Description | Added or Changed in Release | Where Documented |
|---|---|---|---|
| Reset interface configuration to the default configuration | The default interface command allows you to reset an interface to its default configuration. | 6.0(2)U2(1) | Default Interfaces, on page 13<br>Configuring a Default Interface, on page 23 |
| Allow mac-address change for SVI and routed port | You can change the default MAC address of the Layer 3 interface by using the **mac-address** command from the interface configuration mode. | 6.0(2)U2(1) | • Routed Interfaces, on page 36<br>• Configuring an Interface MAC Address, on page 45 |
| Configuring Q-in-Q VLAN Tunnels | Added support for Q-inQ VLAN tunnels. | 6.0(2)U1(1) | Configuring Q-in-Q VLAN Tunnels, on page 139 |



CHAPTER 2

# Configuring Layer 2 Interfaces

This chapter contains the following sections:



# Information About Ethernet Interfaces

The Ethernet ports can operate as standard Ethernet interfaces connected to servers or to a LAN.

The Ethernet interfaces are enabled by default.

## Interface Command

You can enable the various capabilities of the Ethernet interfaces on a per-interface basis using the **interface** command. When you enter the **interface** command, you specify the following information:

The interface numbering convention is extended to support use with a Cisco Nexus Fabric Extender as follows:

switch(config)# **interface ethernet** [*chassis*/]*slot*/*port*

- The chassis ID is an optional entry that you can use to address the ports of a connected Fabric Extender. The chassis ID is configured on a physical Ethernet or EtherChannel interface on the switch to identify the Fabric Extender discovered through the interface. The chassis ID ranges from 100 to 199.

## Unidirectional Link Detection Parameter

The Cisco-proprietary Unidirectional Link Detection (UDLD) protocol allows ports that are connected through fiber optics or copper (for example, Category 5 cabling) Ethernet cables to monitor the physical configuration of the cables and detect when a unidirectional link exists. When the switch detects a unidirectional link, UDLD

shuts down the affected LAN port and alerts the user. Unidirectional links can cause a variety of problems, including spanning tree topology loops.

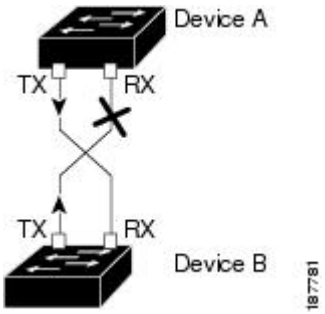
UDLD is a Layer 2 protocol that works with the Layer 1 protocols to determine the physical status of a link. At Layer 1, autonegotiation takes care of physical signaling and fault detection. UDLD performs tasks that autonegotiation cannot perform, such as detecting the identities of neighbors and shutting down misconnected LAN ports. When you enable both autonegotiation and UDLD, Layer 1 and Layer 2 detections work together to prevent physical and logical unidirectional connections and the malfunctioning of other protocols.

A unidirectional link occurs whenever traffic transmitted by the local device over a link is received by the neighbor but traffic transmitted from the neighbor is not received by the local device. If one of the fiber strands in a pair is disconnected, and if autonegotiation is active, the link does not stay up. In this case, the logical link is undetermined, and UDLD does not take any action. If both fibers are working normally at Layer 1, then UDLD at Layer 2 determines whether those fibers are connected correctly and whether traffic is flowing bidirectionally between the correct neighbors. This check cannot be performed by autonegotiation, because autonegotiation operates at Layer 1.

A Cisco Nexus device periodically transmits UDLD frames to neighbor devices on LAN ports with UDLD enabled. If the frames are echoed back within a specific time frame and they lack a specific acknowledgment (echo), the link is flagged as unidirectional and the LAN port is shut down. Devices on both ends of the link must support UDLD in order for the protocol to successfully identify and disable unidirectional links.

The following figure shows an example of a unidirectional link condition. Device B successfully receives traffic from Device A on the port. However, Device A does not receive traffic from Device B on the same port. UDLD detects the problem and disables the port.

***Figure 1: Unidirectional Link***




## Default UDLD Configuration

The following table shows the default UDLD configuration.

***Table 1: UDLD Default Configuration***

| Feature | Default Value |
|---|---|
| UDLD global enable state | Globally disabled |
| UDLD aggressive mode | Disabled |
| UDLD per-port enable state for fiber-optic media | Enabled on all Ethernet fiber-optic LAN ports |

| Feature | Default Value |
| --- | --- |
| UDLD per-port enable state for twisted-pair (copper) media | Enabled |

### UDLD Aggressive and Nonaggressive Modes

UDLD aggressive mode is disabled by default. You can configure UDLD aggressive mode only on point-to-point links between network devices that support UDLD aggressive mode. If UDLD aggressive mode is enabled, when a port on a bidirectional link that has a UDLD neighbor relationship established stops receiving UDLD frames, UDLD tries to reestablish the connection with the neighbor. After eight failed retries, the port is disabled.

To prevent spanning tree loops, nonaggressive UDLD with the default interval of 15 seconds is fast enough to shut down a unidirectional link before a blocking port transitions to the forwarding state (with default spanning tree parameters).

When you enable the UDLD aggressive mode, the following occurs:

- One side of a link has a port stuck (both transmission and receive)
- One side of a link remains up while the other side of the link is down

In these cases, the UDLD aggressive mode disables one of the ports on the link, which prevents traffic from being discarded.

## Interface Speed

Cisco Nexus 3000 Series switches have a number of fixed 10-Gigabit ports; each is equipped with SFP+ interface adapters. Cisco Nexus 3100 Series switches have 32 Quad Same Factor Pluggable (QSFP) ports and 4 SFP+ interface adapters. The default speed for these 32 ports is 40 Gbps.

## 40-Gigabit Ethernet Interface Speed

You can operate QSFP ports as either 40-Gigabit Ethernet or 4 x10-Gigabit Ethernet modes on Cisco Nexus 3132 and Cisco Nexus 3172 switches. By default, there are 32 ports in the 40-Gigabit Ethernet mode. These 40-Gigabit Ethernet ports are numbered in a 2-tuple naming convention. For example, the second 40-Gigabit Ethernet port is numbered as 1/2. The process of changing the configuration from 40-Gigabit Ethernet to 10-Gigabit Ethernet is called breakout and the process of changing the configuration from 10-Gigabit Ethernet to Gigabit Ethernet is called breakin. When you break out a 40-Gigabit Ethernet port into 10-Gigabit Ethernet ports, the resulting ports are numbered using a 3-tuple naming convention. For example, the break-out ports of the second 40-Gigabit Ethernet port are numbered as 1/2/1, 1/2/2, 1/2/3, 1/2/4.

You can break out the 40-Gigabit Ethernet port into four 10-Gigabit Ethernet ports by using the **speed 10000** command and using a splitter cable to connect to multiple peer switches. You can break in four 10-Gigabit Ethernet ports to a 40-Gigabit Ethernet port by using the **speed 40000** command. The configuration change from 40-Gigabit Ethernet to 10-Gigabit Ethernet and from 10-Gigabit Ethernet to 40-Gigabit Ethernet takes effect immediately. You do not need to reload the switch. A QSFP transceiver security check is also performed.

**Note** When you break out from 40-Gigabit Ethernet to 10-Gigabit Ethernet, or break in from 10-Gigabit Ethernet to 40-Gigabit Ethernet, all interface configurations are reset, and the affected ports are administratively unavailable. To make these ports available, use the **no shut** command.

**Note** Starting with Release 6.0(2)U5(1), a new QSFP+ 40-Gb transceiver is now supported on the Cisco Nexus 3000 Series switches. The new QSFP+ (40-Gb) transceiver has a cable that splits into four 10Gb SFP-10G-LR transceivers. To use it, you need the port to be in 4x10G mode. If you are using the breakout cable, you need to run that 40G port in 4x10G mode.

The ability to break out a 40-Gigabit Ethernet port into four 10-Gigabit Ethernet ports and break in four 10-Gigabit Ethernet ports into a 40-Gigabit Ethernet port dynamically allows you to use any of the breakout-capable ports to work in the 40-Gigabit Ethernet or 10-Gigabit Ethernet modes without permanently defining them.

For Cisco Nexus 3132Q switches, when the Ethernet interface 1/1 is in the 40-Gigabit Ethernet mode, the first QSFP port is active. After breakout, when the Ethernet interface 1/1/1-4 is in the 10-Gigabit Ethernet mode, you can choose to use either QSFP ports or SFP+ ports. However, both the first QSFP port and the four SFP+ ports cannot be active at the same time.

# Port Modes

Cisco Nexus 3100 Series switches have various port modes. In Cisco NX-OS Release 6.0(2)U(2)1, only the Cisco Nexus 3132Q switch has port modes that support breakout. Cisco NX-OS Release 6.0(2)U(2)3 introduces breakout port modes for the Cisco Nexus 3172PQ switch.

**Note** Prior to Release 6.0(2)U5(1), the default mode on Cisco Nexus 3132Q and Cisco Nexus 3132CR Series switches used to be Fixed32x40G mode. Starting with Release 6.0(2)U5(1), the default port mode on Cisco Nexus 3132Q and Cisco Nexus 3132CR Series switches after write erase is 32x40G mode.

| Nexus 3100 Series Switches | Ports | Port Modes |
|---|---|---|
| Cisco Nexus 3132Q | 32 x QSFP ports and 4 SFP+ ports | The following port modes support breakout:<br>• 32x40G—This is an oversubscribed port mode. All 32 ports are oversubscribed and the first 24 QSFP ports are break-out capable. You cannot enter the **speed 10000** command on ports 25 through 32. Starting with Release 6.0(2)U5(1), the 32x40G breakout mode is the default port mode.<br>• 26x40G—This is an oversubscribed port mode. Of the 26 ports, 12 ports are nonoversubscribed (cut-through). These ports are 2,4 to 8,14,and 16 to 20. The remaining 14 ports are oversubscribed. All available QSFP ports are break-out capable.<br>• 24x40G—This is the only nonoversubscribed (cut-through) mode. All available QSFP ports are break-out capable.<br>The Fixed32x40G port mode does not support breakout. |
| Cisco Nexus 3172PQ | 6 x QSFP ports and 48 SFP+ ports | The following is the default port mode and supports breakout:<br>• 48x10G+breakout6x40G<br>The following are the fixed port modes that do not support breakout:<br>• 48x10G+6x40G<br>• 72x10G |

# SVI Autostate

The Switch Virtual Interface (SVI) represents a logical interface between the bridging function and the routing function of a VLAN in the device. By default, when a VLAN interface has multiple ports in the VLAN, the SVI goes to the down state when all the ports in the VLAN go down.

Autostate behavior is the operational state of an interface that is governed by the state of the various ports in its corresponding VLAN. An SVI interface on a VLAN comes up when there is at least one port in that vlan that is in STP forwarding state. Similarly, this interface goes down when the last STP forwarding port goes down or goes to another STP state.

By default, Autostate calculation is enabled. You can disable Autostate calculation for an SVI interface and change the default value.

**Note** Nexus 3000 Series switches do not support bridging between two VLANs when an SVI for one VLAN exists on the same device as the bridging link. Traffic coming into the device and bound for the SVI is dropped as a IPv4 discard. This is because the BIA MAC address is shared across VLANs/SVIs with no option to modify the MAC of the SVI.

# Cisco Discovery Protocol

The Cisco Discovery Protocol (CDP) is a device discovery protocol that runs over Layer 2 (the data link layer) on all Cisco-manufactured devices (routers, bridges, access servers, and switches) and allows network management applications to discover Cisco devices that are neighbors of already known devices. With CDP, network management applications can learn the device type and the Simple Network Management Protocol (SNMP) agent address of neighboring devices that are running lower-layer, transparent protocols. This feature enables applications to send SNMP queries to neighboring devices.

CDP runs on all media that support Subnetwork Access Protocol (SNAP). Because CDP runs over the data-link layer only, two systems that support different network-layer protocols can learn about each other.

Each CDP-configured device sends periodic messages to a multicast address, advertising at least one address at which it can receive SNMP messages. The advertisements also contain time-to-live, or holdtime information, which is the length of time a receiving device holds CDP information before discarding it. Each device also listens to the messages sent by other devices to learn about neighboring devices.

The switch supports both CDP Version 1 and Version 2.

## Default CDP Configuration

The following table shows the default CDP configuration.

***Table 2: Default CDP Configuration***

| Feature | Default Setting |
|---|---|
| CDP interface state | Enabled |
| CDP timer (packet update frequency) | 60 seconds |
| CDP holdtime (before discarding) | 180 seconds |
| CDP Version-2 advertisements | Enabled |

# Error-Disabled State

An interface is in the error-disabled (err-disabled) state when the inteface is enabled administratively (using the **no shutdown** command) but disabled at runtime by any process. For example, if UDLD detects a unidirectional link, the interface is shut down at runtime. However, because the interface is administratively enabled, the interface status displays as err-disabled. Once an interface goes into the err-disabled state, you

must manually reenable it or you can configure an automatic timeout recovery value. The err-disabled detection is enabled by default for all causes. The automatic recovery is not configured by default.

When an interface is in the err-disabled state, use the **errdisable detect cause** command to find information about the error.

You can configure the automatic err-disabled recovery timeout for a particular err-disabled cause by changing the time variable.

The **errdisable recovery cause** command provides automatic recovery after 300 seconds. To change the recovery period, use the **errdisable recovery interval** command to specify the timeout period. You can specify 30 to 65535 seconds.

To disable recovery of an interface from the err-disabled state, use the **no errdisable recovery cause** command.

The various options for the **errdisable recover cause** command are as follows:

- all—Enables a timer to recover from all causes.
- bpduguard—Enables a timer to recover from the bridge protocol data unit (BPDU) Guard error-disabled state.
- failed-port-state—Enables a timer to recover from a Spanning Tree Protocol (STP) set port state failure.
- link-flap—Enables a timer to recover from linkstate flapping.
- pause-rate-limit—Enables a timer to recover from the pause rate limit error-disabled state.
- udld—Enables a timer to recover from the Unidirectional Link Detection (UDLD) error-disabled state.
- loopback—Enables a timer to recover from the loopback error-disabled state.

If you do not enable the err-disabled recovery for the cause, the interface stays in the err-disabled state until you enter the **shutdown** and **no shutdown** commands. If the recovery is enabled for a cause, the interface is brought out of the err-disabled state and allowed to retry operation once all the causes have timed out. Use the **show interface status err-disabled** command to display the reason behind the error.

# Default Interfaces

You can use the default interface feature to clear the configured parameters for both physical and logical interfaces such as the Ethernet, loopback, management, VLAN, and the port-channel interface.

# Debounce Timer Parameters

The debounce timer delays notification of a link change, which can decrease traffic loss due to network reconfiguration. You can configure the debounce timer separately for each Ethernet port and specify the delay time in milliseconds. The delay time can range from 0 milliseconds to 5000 milliseconds. By default, this parameter is set for 100 milliseconds, which results in the debounce timer not running. When this parameter is set to 0 milliseconds, the debounce timer is disabled.

**Caution** Enabling the debounce timer causes the link-down detections to be delayed, which results in a loss of traffic during the debounce period. This situation might affect the convergence and reconvergence of some Layer 2 and Layer 3 protocols.

# MTU Configuration

The Cisco Nexus device switch does not fragment frames. As a result, the switch cannot have two ports in the same Layer 2 domain with different maximum transmission units (MTUs). A per-physical Ethernet interface MTU is not supported. Instead, the MTU is set according to the QoS classes. You modify the MTU by setting class and policy maps.

**Note** When you show the interface settings, a default MTU of 1500 is displayed for physical Ethernet interfaces.

# Downlink Delay

You can operationally enable uplink SFP+ ports before downlink RJ-45 ports after a reload on a Cisco Nexus 3048 switch. You must delay enabling the RJ-45 ports in the hardware until the SFP+ ports are enabled.

You can configure a timer that during reload enables the downlink RJ-45 ports in hardware only after the specified timeout. This process allows the uplink SFP+ ports to be operational first. The timer is enabled in the hardware for only those ports that are admin-enable.

Downlink delay is disabled by default and must be explicitly enabled. When enabled, if the delay timer is not specified, it is set for a default delay of 20 seconds.

# Default Physical Ethernet Settings

The following table lists the default settings for all physical Ethernet interfaces:

| Parameter | Default Setting |
|---|---|
| Duplex | Auto (full-duplex) |
| Encapsulation | ARPA |
| MTU[1] | 1500 bytes |
| Port Mode | Access |
| Speed | Auto (10000) |

1 MTU cannot be changed per-physical Ethernet interface. You modify MTU by selecting maps of QoS classes.

# Configuring Ethernet Interfaces

## Configuring the UDLD Mode

You can configure normal or aggressive unidirectional link detection (UDLD) modes for Ethernet interfaces on devices configured to run UDLD. Before you can enable a UDLD mode for an interface, you must make sure that UDLD is already enabled on the device that includes the interface. UDLD must also be enabled on the other linked interface and its device.

To use the normal UDLD mode, you must configure one of the ports for normal mode and configure the other port for the normal or aggressive mode. To use the aggressive UDLD mode, you must configure both ports for the aggressive mode.

**Note** Before you begin, UDLD must be enabled for the other linked port and its device.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **feature udld** | Enables UDLD for the device. |
| **Step 3** | switch(config)# **no feature udld** | Disables UDLD for the device. |
| **Step 4** | switch(config)# **show udld global** | Displays the UDLD status for the device. |
| **Step 5** | switch(config)# **interface** *type slot*/*port* | Specifies an interface to configure, and enters interface configuration mode. |
| **Step 6** | switch(config-if)# **udld** {**enable** \| **disable** \| **aggressive**} | Enables the normal UDLD mode, disables UDLD, or enables the aggressive UDLD mode. |
| **Step 7** | switch(config-if)# **show udld** *interface* | Displays the UDLD status for the interface. |

This example shows how to enable UDLD for the switch:

```
switch# configure terminal
switch(config)# feature udld
```

This example shows how to enable the normal UDLD mode for an Ethernet port:

```
switch# configure terminal
switch(config)# interface ethernet 1/4
switch(config-if)# udld enable
```

This example shows how to enable the aggressive UDLD mode for an Ethernet port:

```
switch# configure terminal
```

```
switch(config)# interface ethernet 1/4
switch(config-if)# udld aggressive
```

This example shows how to disable UDLD for an Ethernet port:

```
switch# configure terminal
switch(config)# interface ethernet 1/4
switch(config-if)# udld disable
```

This example shows how to disable UDLD for the switch:

```
switch# configure terminal
switch(config)# no feature udld
```

# Triggering the Link State Consistency Checker

You can manually trigger the link state consistency checker to compare the hardware and software link status of an interface and display the results. To manually trigger the link state consistency checker and display the results, use the following command in any mode:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **show consistency-checker link-state module** *slot* | Starts a link state consistency check on the specified module and displays its results. |

This example shows how to trigger a Link State consistency check and display its results:

```
switch# show consistency-checker link-state module 1
Link State Checks: Link state only
Consistency Check: FAILED
No inconsistencies found for:
   Ethernet1/1
   Ethernet1/2
   Ethernet1/3
   Ethernet1/4
   Ethernet1/5
   Ethernet1/6
   Ethernet1/7
   Ethernet1/8
   Ethernet1/9
   Ethernet1/10
   Ethernet1/12
   Ethernet1/13
   Ethernet1/14
   Ethernet1/15
Inconsistencies found for following interfaces:
   Ethernet1/11
```

# Changing an Interface Port Mode

You can configure a Quad small form-factor pluggable (QSFP+) port by using the **hardware profile** *portmode* command. To restore the defaults, use the **no** form of these commands. The Cisco Nexus 3172PQ switch has 48x10g+breakout6x40g as the default port mode.

**Note** Using the CVR-QSFP-SFP10G adapter does not work with the twinax cables in 40G ports on the Cisco Nexus 3000 Series platforms. You have to first configure the port for breakout. You can then use the first port in the breakout.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **copy running-config bootflash**: *my-config.cfg* | Copies the running configuration to the bootflash. You can use this file to configure your device later. |
| **Step 3** | switch(config)# **write erase** | Removes all the interface configurations. |
| **Step 4** | switch(config)# **reload** | Reloads the Cisco NX-OS software. |
| **Step 5** | switch(config)# [**no**] **hardware profile portmode** *portmode* | Changes the interface port mode. |
| **Step 6** | switch(config)# **hardware profile portmode** *portmode* **2-tuple** | (Optional)<br>Displays the port names in 2-tuple mode instead of the default 3-tuple convention mode. |
| **Step 7** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |
| **Step 8** | switch(config)# **reload** | Reloads the Cisco NX-OS software.<br>Manually apply all the interface configuration. You can refer to the configuration file that you saved earlier.<br>**Note** The interface numbering changes if the ports are changed from 40G mode to 4x10G mode or vice versa. |

This example shows how to change the port mode to 48x10g+breakout6x40g for QSFP+ ports:

```
switch# configure terminal
switch(config)# copy running-config bootflash:my-config.cfg
switch(config)# write erase
switch(config)# reload
WARNING: This command will reboot the system
Do you want to continue? (y/n) [n] y
switch(config)# hardware profile portmode 48x10g+breakout6x40g
Warning: This command will take effect only after saving the configuration and reload!
Port configurations could get lost when port mode is changed!
switch(config)# copy running-config startup-config
switch(config)# reload
WARNING: This command will reboot the system
Do you want to continue? (y/n) [n] y
```

This example shows how to change the port mode to 48x10g+4x40g for QSFP+ ports:

```
switch# configure terminal
switch(config)# copy running-config bootflash:my-config.cfg
switch(config)# write erase
switch(config)# reload
WARNING: This command will reboot the system
Do you want to continue? (y/n) [n] y
switch(config)# hardware profile portmode 48x10g+4x40g
Warning: This command will take effect only after saving the configuration and reload!
Port configurations could get lost when port mode is changed!
switch(config)# copy running-config startup-config
switch(config)# reload
WARNING: This command will reboot the system
Do you want to continue? (y/n) [n] y
```

This example shows how to change the port mode to 48x10g+4x40g for QSFP+ ports and verify the changes:

```
switch# configure terminal
switch(config)# hardware profile portmode 48x10g+4x40g
Warning: This command will take effect only after saving the configuration and r
eload! Port configurations could get lost when port mode is changed!
switch(config)# show running-config
!Command: show running-config
!Time: Thu Aug 25 07:39:37 2011
version 5.0(3)U2(1)
feature telnet
no feature ssh
feature lldp
username admin password 5 $1$OOV4MdOM$BAB5RkD22YanT4empqqSM0 role network-admin
ip domain-lookup
switchname BLR-QG-5
ip access-list my-acl
10 deny ip any 10.0.0.1/32
20 deny ip 10.1.1.1/32 any
class-map type control-plane match-any copp-arp
class-map type control-plane match-any copp-bpdu
:
:
control-plane
service-policy input copp-system-policy
hardware profile tcam region arpacl 128
hardware profile tcam region ifacl 256
hardware profile tcam region racl 256
hardware profile tcam region vacl 512
hardware profile portmode 48x10G+4x40G
snmp-server user admin network-admin auth md5 0xdd1d21ee42e93106836cdefd1a60e062
<--Output truncated-->
switch#
```

This example shows how to restore the default port mode for QSFP+ ports:

```
switch# configure terminal
switch(config)# no hardware profile portmode
Warning: This command will take effect only after saving the configuration and r
eload! Port configurations could get lost when port mode is changed!
switch(config)#
```

# Configuring the Interface Speed

**Note** If the interface and transceiver speed is mismatched, the SFP validation failed message is displayed when you enter the **show interface ethernet** *slot*/*port* command. For example, if you insert a 1-Gigabit SFP transceiver into a port without configuring the **speed 1000** command, you will get this error. By default, all ports are 10 Gbps.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface** *type slot/port* | Enters interface configuration mode for the specified interface. This interface must have a 1-Gigabit Ethernet SFP transceiver inserted into it. |
| **Step 3** | switch(config-if)# **speed** *speed* | Sets the speed on the interface.<br>This command can only be applied to a physical Ethernet interface. The *speed* argument can be set to one of the following:<br>• 10 Mbps<br>• 100 Mbps<br>• 1 Gbps<br>• 10 Gbps<br>• automatic |

This example shows how to set the speed for a 1-Gigabit Ethernet port:

```
switch# configure terminal
switch(config)# interface ethernet 1/4
switch(config-if)# speed 1000
```

# Configuring Break-Out 10-Gigabit Interface Speed Ports

By default, all ports on Cisco Nexus 3132 switches are 40-Gigabit Ethernet. You can break out a 40-Gigabit Ethernet port to four x10-Gigabit Ethernet ports.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface** *type slot/port-range* | Enters interface configuration mode for the specified interface.<br>**Note** Interface range is not supported for 40-Gigabit Ethernet interfaces. For example, Eth 1/2-5 is not supported. |
| **Step 3** | switch(config-if)# **speed 10000** | Sets the speed on the interface to 10-Gigabit per second. |

This example shows how to set the speed to 10-Gigabit per second on Ethernet interface 1/2:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# speed 10000
```

# Configuring Break-In 40-Gigabit Ethernet Interface Speed Ports

You can break in four x 10-Gigabit Ethernet ports to a 40-Gigabit Ethernet port.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface** *type slot/port* | Enters interface configuration mode for the specified interface.<br>**Note** The Interface range is supported for 10-Gigabit Ethernet interfaces. For example, Eth 1/2/1-4 is supported. |
| **Step 3** | switch(config-if)# **speed 40000** | Sets the speed on the interface to 40 Gbps. |

This example shows how to set the speed to 40 Gbps on Ethernet interface 1/2/1:

```
switch# configure terminal
switch(config)# interface ethernet 1/2/1
switch(config-if)# speed 40000
```

# Switching Between QSFP and SFP+ Ports

When you break out ports into the 10-GbE mode, you can switch between the first QSFP port and SFP+ ports 1 to 4. Either the first QSFP port or the four SFP+ ports can be active at any time. QSFP is the default port with an interface speed of 40 Gbps.

When the first QSFP port is in the 40-GbE mode, you cannot switch the port to four SFP+ ports and the first QSFP port will be active until you break out the port into the 10-GbE mode. This is because SFP+ ports do not support the 40-GbE mode.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **[no] hardware profile front portmode qsfp** \| **sfp-plus** | Activates the specified port mode.<br>• **qsfp—**The front panel QSFP port is active<br>• **sfp-plus—**The front panel SFP+ ports 1 to 4 are active<br>The **no** form of this command activates the QSFP port. |

| | Command or Action | Purpose |
|---|---|---|
| | | **Note** If the first QSFP port speed is 40 Gbps, this command will run, but the SFP+ ports will not become active until after the speed is changed to 10 Gbps.<br>**Note** The SFP+ ports are active only after configuring the port in 10Gx4 breakout mode. |
| **Step 3** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to make the SFP+ ports active:

```
switch# configure terminal
switch(config)# hardware profile front portmode sfp-plus
switch(config)# interface Ethernet 1/1/1
switch(config-if)# speed 10000
switch(config-if)# exit
switch(config-if)# show interface brief
switch(config)# copy running-config startup-config
```

This example shows how to make the QSFP port active:

```
switch# configure terminal
switch(config)# no hardware profile front portmode
switch(config)# copy running-config startup-config
```

# Disabling Link Negotiation

By default, auto-negotiation is enabled on all 1G SFP+ and 40G QSFP ports and it is disabled on 10G SFP+ ports. Auto-negotiation is by default enabled on all 1G and 10G Base-T ports. It cannot be disabled on 1G and 10G Base-T ports.

This command is equivalent to the Cisco IOS **speed non-negotiate** command.

Starting with Release 6.0(2)U5(1), you can disable auto-negotiation on all 40G interfaces. A new CLI command **no system default interface 40g auto-negotation** is introduced to disable auto-negotiation across all the 40G interfaces. The new CLI command is only effective on the 40G interfaces and it does not have any effect on 1G or10G interfaces. For CR4 cables, the auto-negotiation configuration has to be identical at both the end devices for the link to come up.

**Note** The auto-negotiation configuration is not applicable on 10-Gigabit Ethernet ports. When auto-negotiation is configured on a 10-Gigabit port, the following error message is displayed:

```
ERROR: Ethernet1/40: Configuration does not match the port capability
```

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | switch(config)# **interface ethernet** *slot*/*port* | Selects the interface and enters interface mode. |
| **Step 3** | switch(config-if)# **no negotiate auto** | Disables link negotiation on the selected Ethernet interface (1-Gigabit port). |
| **Step 4** | switch(config-if)# **negotiate auto** | (Optional)<br>Enables link negotiation on the selected Ethernet interface. The default for 1-Gigabit Ethernet ports is enabled.<br>**Note** This command is not applicable for 10GBASE-T ports. It should not be used on 10-GBASE-T ports. |

This example shows how to disable auto-negotiation on a specified Ethernet interface (1-Gigabit port):

```
switch# configure terminal
switch(config)# interface ethernet 1/1
switch(config-if)# no negotiate auto
switch(config-if)#
```

This example shows how to enable auto-negotiation on a specified Ethernet interface (1-Gigabit port):

```
switch# configure terminal
switch(config)# interface ethernet 1/5
switch(config-if)# negotiate auto
switch(config-if)#
```

The following example displays the new CLI introduced for global knob for auto-negotiation disable. The system default configuration is **system default interface 40g auto-negotiation**.

```
switch# configure terminal
(config)# no system default interface 40g auto-negotiation
(config)# sh running-config | grep auto-neg
no system default interface 40g auto-negotiation
(config)# sh interface eth1/19 brief
Eth1/19    1   eth  access up      none    40G(D) --
(config)# sh running-config interface eth1/19 all | grep auto
no negotiate auto
(config)#
```

# Disabling SVI Autostate

You can configure a SVI to remain active even if no interfaces are up in the corresponding VLAN. This enhancement is called Autostate Disable.

When you enable or disable autostate behavior, it is applied to all the SVIs in the switch unless you configure autostate per SVI .

**Note** Autostate behavior is enabled by default.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **feature interface-vlan** | Enables the interface-vlan feature. |
| **Step 3** | switch(config)# **system default interface-vlan** [**no**] **autostate** | Configures the system to enable or disable the Autostate default behavior. |
| **Step 4** | switch(config)# **interface vlan** *interface-vlan-number* | (Optional)<br>Creates a VLAN interface. The number range is from 1 to 4094. |
| **Step 5** | switch(config-if)# [**no**] **autostate** | (Optional)<br>Enables or disables Autostate behavior per SVI. |
| **Step 6** | switch(config)# **show interface-vlan** *interface-vlan* | (Optional)<br>Displays the enabled or disabled Autostate behavior of the SVI. |
| **Step 7** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to disable the systems Autostate default for all the SVIs on the switch:

```
switch# configure terminal
switch(config)# feature interface-vlan
switch(config)# system default interface-vlan no autostate
switch(config)# interface vlan 50
switch(config-if)# no autostate
switch(config)# copy running-config startup-config
```

This example shows how to enable the systems autostate configuration:

```
switch(config)# show interface-vlan 2
Vlan2 is down, line protocol is down, autostate enabled
Hardware is EtherSVI, address is 547f.ee40.a17c
MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec
```

# Configuring a Default Interface

The default interface feature allows you to clear the existing configuration of multiple interfaces such as Ethernet, loopback, management, VLAN, and port-channel interfaces. All user configuration under a specified interface will be deleted.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **default interface** *type interface number* | Deletes the configuration of the interface and restores the default configuration. The following are the supported interfaces:<br>• **ethernet**<br>• **loopback**<br>• **mgmt**<br>• **port-channel**<br>• **vlan** |
| **Step 3** | switch(config)# **exit** | Exits global configuration mode. |

This example shows how to delete the configuration of an Ethernet interface and revert it to its default configuration:

```
switch# configure terminal
switch(config)# default interface ethernet 1/3
.......Done
switch(config)# exit
```

# Configuring the CDP Characteristics

You can configure the frequency of Cisco Discovery Protocol (CDP) updates, the amount of time to hold the information before discarding it, and whether or not to send Version-2 advertisements.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# [**no**] **cdp advertise** {**v1** \| **v2** } | (Optional)<br>Configures the version to use to send CDP advertisements. Version-2 is the default state.<br>Use the **no** form of the command to return to its default setting. |
| **Step 3** | switch(config)# [**no**] **cdp format device-id** {**mac-address** \| **serial-number** \| **system-name**} | (Optional)<br>Configures the format of the CDP device ID. The default is the system name, which can be expressed as a fully qualified domain name. |

| | Command or Action | Purpose |
|---|---|---|
| | | Use the **no** form of the command to return to its default setting. |
| **Step 4** | switch(config)# [**no**] **cdp holdtime** *seconds* | (Optional)<br>Specifies the amount of time a receiving device should hold the information sent by your device before discarding it. The range is 10 to 255 seconds; the default is 180 seconds.<br>Use the **no** form of the command to return to its default setting. |
| **Step 5** | switch(config)# [**no**] **cdp timer** *seconds* | (Optional)<br>Sets the transmission frequency of CDP updates in seconds. The range is 5 to 254; the default is 60 seconds.<br>Use the **no** form of the command to return to its default setting. |

This example shows how to configure CDP characteristics:

```
switch# configure terminal
switch(config)# cdp timer 50
switch(config)# cdp holdtime 120
switch(config)# cdp advertise v2
```

# Enabling or Disabling CDP

You can enable or disable CDP for Ethernet interfaces. This protocol works only when you have it enabled on both interfaces on the same link.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface** *type slot*/*port* | Enters interface configuration mode for the specified interface. |
| **Step 3** | switch(config-if)# **cdp enable** | Enables CDP for the interface.<br>To work correctly, this parameter must be enabled for both interfaces on the same link. |
| **Step 4** | switch(config-if)# **no cdp enable** | Disables CDP for the interface. |

This example shows how to enable CDP for an Ethernet port:

```
switch# configure terminal
switch(config)# interface ethernet 1/4
```

```
switch(config-if)# cdp enable
```

This command can only be applied to a physical Ethernet interface.

# Enabling the Error-Disabled Detection

You can enable error-disable (err-disabled) detection in an application. As a result, when a cause is detected on an interface, the interface is placed in an err-disabled state, which is an operational state that is similar to the link-down state.

**Note** Base ports in Cisco Nexus 5500 never get error disabled due to pause rate-limit like in the Cisco Nexus 5020 or 5010 switch.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **errdisable detect cause** {*all* \| *link-flap* \| *loopback*} | Specifies a condition under which to place the interface in an err-disabled state. The default is enabled. |
| **Step 3** | switch(config)# **shutdown** | Brings the interface down administratively. To manually recover the interface from the err-disabled state, enter this command first. |
| **Step 4** | switch(config)# **no shutdown** | Brings the interface up administratively and enables the interface to recover manually from the err-disabled state. |
| **Step 5** | switch(config)# **show interface status err-disabled** | Displays information about err-disabled interfaces. |
| **Step 6** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to enable the err-disabled detection in all cases:

```
switch# configure terminal
switch(config)# errdisable detect cause all
switch(config)# shutdown
switch(config)# no shutdown
switch(config)# show interface status err-disabled
switch(config)# copy running-config startup-config
```

# Enabling the Error-Disabled Recovery

You can specify the application to bring the interface out of the error-disabled (err-disabled) state and retry coming up. It retries after 300 seconds, unless you configure the recovery timer (see the **errdisable recovery interval** command).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **errdisable recovery cause** {*all* \| *udld* \| *bpduguard* \| *link-flap* \| *failed-port-state* \| *pause-rate-limit* \| *loopback*} | Specifies a condition under which the interface automatically recovers from the err-disabled state, and the device retries bringing the interface up. The device waits 300 seconds to retry. The default is disabled. |
| **Step 3** | switch(config)# **show interface status err-disabled** | Displays information about err-disabled interfaces. |
| **Step 4** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to enable err-disabled recovery under all conditions:

```
switch# configure terminal
switch(config)# errdisable recovery cause loopback
switch(config)# show interface status err-disabled
switch(config)# copy running-config startup-config
```

# Configuring the Error-Disabled Recovery Interval

You can use this procedure to configure the err-disabled recovery timer value. The range is from 30 to 65535 seconds. The default is 300 seconds.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **errdisable recovery interval** *interval* | Specifies the interval for the interface to recover from the err-disabled state. The range is from 30 to 65535 seconds. The default is 300 seconds. |
| **Step 3** | switch(config)# **show interface status err-disabled** | Displays information about err-disabled interfaces. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to enable err-disabled recovery under all conditions:

```
switch# configure terminal
switch(config)# errdisable recovery interval 32
switch(config)# show interface status err-disabled
switch(config)# copy running-config startup-config
```

# Disabling the Error-Disabled Recovery

You can disable recovery of an interface from the err-disabled state.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **no errdisable recovery cause** {*all* \| *udld* \| *bpduguard* \| *link-flap* \| *failed-port-state* \| *pause-rate-limit* \| *loopback*} | Specifies a condition under which the interface reverts back to the default err-disabled state. |
| **Step 3** | switch(config)# **show interface status err-disabled** | (Optional)<br>Displays information about err-disabled interfaces. |
| **Step 4** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to disable err-disabled recovery:

```
switch# configure terminal
switch(config)# no errdisable recovery cause loopback
switch(config)# show interface status err-disabled
switch(config)# copy running-config startup-config
```

# Configuring the Debounce Timer

You can enable the debounce timer for Ethernet ports by specifying a debounce time, in milliseconds (ms), or disable the timer by specifying a debounce time of 0. By default, the debounce timer is set to 100 ms, which results in the debounce timer not running.

You can show the debounce times for all of the Ethernet ports by using the **show interface debounce** command.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface** *type slot/port* | Enters interface configuration mode for the specified interface. |
| **Step 3** | switch(config-if)# **link debounce time** *milliseconds* | Enables the debounce timer for the amount of time (1 to 5000 ms) specified.<br>Disables the debounce timer if you specify 0 milliseconds. |

This example shows how to enable the debounce timer and set the debounce time to 1000 ms for an Ethernet interface:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# link debounce time 1000
```

This example shows how to disable the debounce timer for an Ethernet interface:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# link debounce time 0
```

# Configuring the Description Parameter

You can provide textual interface descriptions for the Ethernet ports.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface** *type slot/port* | Enters interface configuration mode for the specified interface. |
| **Step 3** | switch(config-if)# **description** *test* | Specifies the description for the interface. |

This example shows how to set the interface description to Server 3 interface:

```
switch# configure terminal
switch(config)# interface ethernet 1/3
switch(config-if)# description Server 3 Interface
```

# Disabling and Restarting Ethernet Interfaces

You can shut down and restart an Ethernet interface. This action disables all of the interface functions and marks the interface as being down on all monitoring displays. This information is communicated to other network servers through all dynamic routing protocols. When shut down, the interface is not included in any routing updates.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface** *type slot*/*port* | Enters interface configuration mode for the specified interface. |
| **Step 3** | switch(config-if)# **shutdown** | Disables the interface. |
| **Step 4** | switch(config-if)# **no shutdown** | Restarts the interface. |

This example shows how to disable an Ethernet port:

```
switch# configure terminal
switch(config)# interface ethernet 1/4
switch(config-if)# shutdown
```

This example shows how to restart an Ethernet interface:

```
switch# configure terminal
switch(config)# interface ethernet 1/4
switch(config-if)# no shutdown
```

# Configuring Downlink Delay

You can operationally enable uplink SFP+ ports before downlink RJ-45 ports after a reload on a Cisco Nexus 3048 switch by delaying enabling the RJ-45 ports in the hardware until the SFP+ ports are enabled.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **downlink delay enable** \| **disable** [**timeout** *time-out*] | Enables or disables downlink delay and configures the timeout. |

This example shows how to enable downlink delay and configure the delay timeout on the switch:

```
switch# configure terminal
switch(config)# downlink delay enable timeout 45
```

# Displaying Interface Information

To view configuration information about the defined interfaces, perform one of these tasks:

| Command | Purpose |
|---|---|
| switch# **show interface** *type slot*/*port* | Displays the detailed configuration of the specified interface. |
| switch# **show interface** *type slot*/*port* **capabilities** | Displays detailed information about the capabilities of the specified interface. This option is available only for physical interfaces. |
| switch# **show interface** *type slot*/*port* **transceiver** | Displays detailed information about the transceiver connected to the specified interface. This option is available only for physical interfaces. |
| switch# **show interface brief** | Displays the status of all interfaces. |
| switch# **show interface flowcontrol** | Displays the detailed listing of the flow control settings on all interfaces. |

The **show interface** command is invoked from EXEC mode and displays the interface configurations. Without any arguments, this command displays the information for all the configured interfaces in the switch.

This example shows how to display the physical Ethernet interface:

```
switch# show interface ethernet 1/1
  Ethernet1/1 is up
  Hardware is 1000/10000 Ethernet, address is 000d.eca3.5f08 (bia 000d.eca3.5f08)
  MTU 1500 bytes, BW 10000000 Kbit, DLY 10 usec,
     reliability 255/255, txload 190/255, rxload 192/255
  Encapsulation ARPA
  Port mode is trunk
  full-duplex, 10 Gb/s, media type is 1/10g
  Input flow-control is off, output flow-control is off
  Auto-mdix is turned on
  Rate mode is dedicated
  Switchport monitor is off
  Last clearing of "show interface" counters never
  5 minute input rate 942201806 bytes/sec, 14721892 packets/sec
  5 minute output rate 935840313 bytes/sec, 14622492 packets/sec
  Rx
    129141483840 input packets 0 unicast packets 129141483847 multicast packets
    0 broadcast packets 0 jumbo packets 0 storm suppression packets
    8265054965824 bytes
    0 No buffer 0 runt 0 Overrun
    0 crc 0 Ignored 0 Bad etype drop
    0 Bad proto drop
  Tx
```

```
    119038487241 output packets 119038487245 multicast packets
   0 broadcast packets 0 jumbo packets
    7618463256471 bytes
    0 output CRC 0 ecc
    0 underrun 0 if down drop     0 output error 0 collision 0 deferred
    0 late collision 0 lost carrier 0 no carrier
    0 babble
    0 Rx pause 8031547972 Tx pause 0 reset
```

This example shows how to display the physical Ethernet capabilities:

```
switch# show interface ethernet 1/1 capabilities
Ethernet1/1
  Model:                734510033
  Type:                 10Gbase-(unknown)
  Speed:                1000,10000
  Duplex:               full
  Trunk encap. type:    802.1Q
  Channel:              yes
  Broadcast suppression: percentage(0-100)
  Flowcontrol:          rx-(off/on),tx-(off/on)
  Rate mode:            none
  QOS scheduling:       rx-(6q1t),tx-(1p6q0t)
  CoS rewrite:          no
  ToS rewrite:          no
  SPAN:                 yes
  UDLD:                 yes

  MDIX:                 no
  FEX Fabric:           yes
```

This example shows how to display the physical Ethernet transceiver:

```
switch# show interface ethernet 1/1 transceiver
Ethernet1/1
    sfp is present
    name is CISCO-EXCELIGHT
    part number is SPP5101SR-C1
    revision is A
    serial number is ECL120901AV
    nominal bitrate is 10300 MBits/sec
    Link length supported for 50/125mm fiber is 82 m(s)
    Link length supported for 62.5/125mm fiber is 26 m(s)
    cisco id is --
    cisco extended id number is 4
```

This example shows how to display a brief interface status (some of the output has been removed for brevity):

```
switch# show interface brief

--------------------------------------------------------------------------------
Ethernet      VLAN   Type Mode   Status  Reason                   Speed     Port
Interface                                                                  Ch #
--------------------------------------------------------------------------------
Eth1/1        200    eth  trunk  up      none                       10G(D) --
Eth1/2        1      eth  trunk  up      none                       10G(D) --
Eth1/3        300    eth  access down    SFP not inserted           10G(D) --
Eth1/4        300    eth  access down    SFP not inserted           10G(D) --
Eth1/5        300    eth  access down    Link not connected        1000(D) --
Eth1/6        20     eth  access down    Link not connected         10G(D) --
Eth1/7        300    eth  access down    SFP not inserted           10G(D) --
...
```

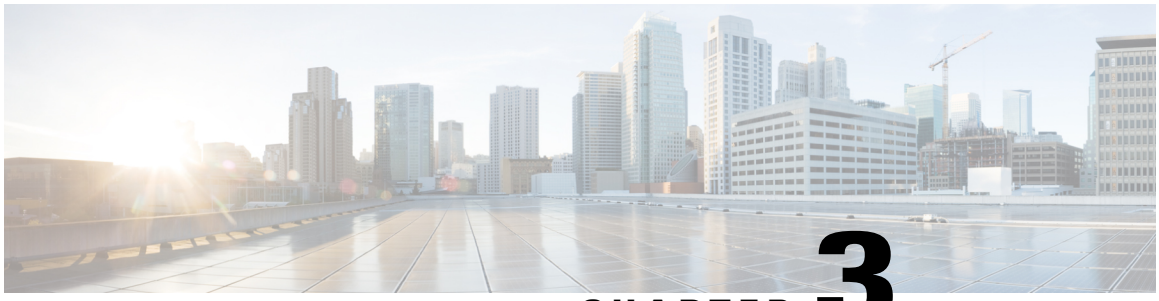
This example shows how to display the CDP neighbors:

```
switch# show cdp neighbors
Capability Codes: R - Router, T - Trans-Bridge, B - Source-Route-Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater,
                  V - VoIP-Phone, D - Remotely-Managed-Device,
                  s - Supports-STP-Dispute
Device ID              Local Intrfce   Hldtme  Capability  Platform      Port ID
d13-dist-1               mgmt0           148     S I      WS-C2960-24TC  Fas0/9
```

```
n5k(FLC12080012)         Eth1/5          8       S I s    N5K-C5020P-BA  Eth1/5
```

# MIBs for Layer 2 Interfaces

| MIB | MIB Link |
|---|---|
| IF-MIB | To locate and download MIBs, go to the following URL:<br>http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml |
| MAU-MIB<br>Limited support includes only the following MIB Objects:<br>• ifMauType (Read-only) GET<br>• ifMauAutoNegSupported (Read-only) GET<br>• ifMauTypeListBits (Read-only) GET<br>• ifMauDefaultType (Read-write) GET-SET<br>• ifMauAutoNegAdminStatus (Read-write) GET-SET<br>• ifMauAutoNegCapabilityBits (Read-only) GET<br>• ifMauAutoNegAdvertisedBits (Read-write) GET-SET | |



CHAPTER 3

# Configuring Layer 3 Interfaces

This chapter contains the following sections:



# Information About Layer 3 Interfaces

Layer 3 interfaces forward packets to another device using static or dynamic routing protocols. You can use Layer 3 interfaces for IP routing and inter-VLAN routing of Layer 2 traffic.

# Routed Interfaces

You can configure a port as a Layer 2 interface or a Layer 3 interface. A routed interface is a physical port that can route IP traffic to another device. A routed interface is a Layer 3 interface only and does not support Layer 2 protocols, such as the Spanning Tree Protocol (STP).

All Ethernet ports are Layer 2 (switchports) by default. You can change this default behavior using the **no switchport** command from interface configuration mode. To change multiple ports at one time, you can specify a range of interfaces and then apply the **no switchport** command.

You can assign an IP address to the port, enable routing, and assign routing protocol characteristics to this routed interface.

You can assign a static MAC address to a Layer 3 interface. The default MAC address for a Layer 3 interface is the MAC address of the virtual device context (VDC) that is associated with it. You can change the default MAC address of the Layer 3 interface by using the **mac-address** command from the interface configuration mode. A static MAC address can be configured on SVI, Layer 3 interfaces, port channels, Layer 3 subinterfaces, and tunnel interfaces. You can also configure static MAC addresses on a range of ports and port channels. However, all ports must be in Layer 3. Even if one port in the range of ports is in Layer 2, the command is rejected and an error message appears.  For information on configuring MAC addresses, see the Layer 2 Switching Configuration Guide for your device.

You can also create a Layer 3 port channel from routed interfaces.

Routed interfaces and subinterfaces support exponentially decayed rate counters. Cisco NX-OS tracks the following statistics with these averaging counters:

- Input packets/sec
- Output packets/sec
- Input bytes/sec
- Output bytes/sec

# Subinterfaces

You can create virtual subinterfaces on a parent interface configured as a Layer 3 interface. A parent interface can be a physical port or a port channel.

Subinterfaces divide the parent interface into two or more virtual interfaces on which you can assign unique Layer 3 parameters such as IP addresses and dynamic routing protocols. The IP address for each subinterface should be in a different subnet from any other subinterface on the parent interface.

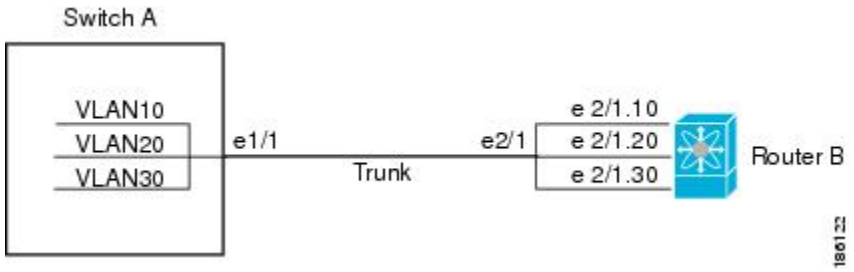
You create a subinterface with a name that consists of the parent interface name (for example, Ethernet 2/1) followed by a period and then by a number that is unique for that subinterface. For example, you could create a subinterface for Ethernet interface 2/1 named Ethernet 2/1.1 where .1 indicates the subinterface.

Cisco NX-OS enables subinterfaces when the parent interface is enabled. You can shut down a subinterface independent of shutting down the parent interface. If you shut down the parent interface, Cisco NX-OS shuts down all associated subinterfaces as well.

One use of subinterfaces is to provide unique Layer 3 interfaces to each VLAN that is supported by the parent interface. In this scenario, the parent interface connects to a Layer 2 trunking port on another device. You configure a subinterface and associate the subinterface to a VLAN ID using 802.1Q trunking.

The following figure shows a trunking port from a switch that connects to router B on interface E 2/1. This interface contains three subinterfaces that are associated with each of the three VLANs that are carried by the trunking port.

***Figure 2: Subinterfaces for VLANs***




# VLAN Interfaces

A VLAN interface or a switch virtual interface (SVI) is a virtual routed interface that connects a VLAN on the device to the Layer 3 router engine on the same device. Only one VLAN interface can be associated with a VLAN, but you need to configure a VLAN interface for a VLAN only when you want to route between VLANs or to provide IP host connectivity to the device through a virtual routing and forwarding (VRF) instance that is not the management VRF. When you enable VLAN interface creation, Cisco NX-OS creates a VLAN interface for the default VLAN (VLAN 1) to permit remote switch administration.

You must enable the VLAN network interface feature before you can configure it. The system automatically takes a checkpoint prior to disabling the feature, and you can roll back to this checkpoint. For information about rollbacks and checkpoints, see the System Management Configuration Guide for your device.

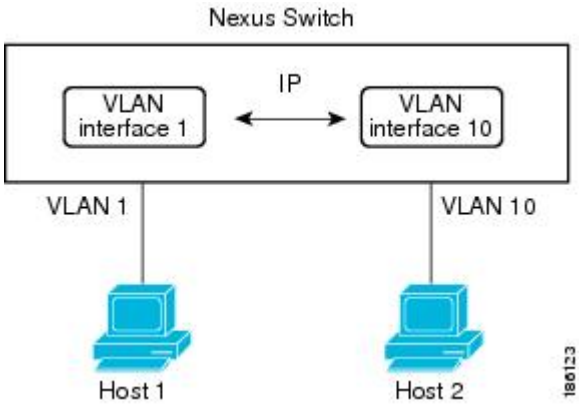
**Note** You cannot delete the VLAN interface for VLAN 1.

You can route across VLAN interfaces to provide Layer 3 inter-VLAN routing by configuring a VLAN interface for each VLAN that you want to route traffic to and assigning an IP address on the VLAN interface. For more information on IP addresses and IP routing, see the Unicast Routing Configuration Guide for your device.

The following figure shows two hosts connected to two VLANs on a device. You can configure VLAN interfaces for each VLAN that allows Host 1 to communicate with Host 2 using IP routing between the VLANs.

VLAN 1 communicates at Layer 3 over VLAN interface 1and VLAN 10 communicates at Layer 3 over VLAN interface 10.

***Figure 3: Connecting Two VLANs with VLAN Interfaces***




## Loopback Interfaces

A loopback interface is a virtual interface with a single endpoint that is always up. Any packet that is transmitted over a loopback interface is immediately received by this interface. Loopback interfaces emulate a physical interface.

You can use loopback interfaces for performance analysis, testing, and local communications. Loopback interfaces can act as a termination address for routing protocol sessions. This loopback configuration allows routing protocol sessions to stay up even if some of the outbound interfaces are down.

## Tunnel Interfaces

Cisco NX-OS supports tunnel interfaces as IP tunnels. IP tunnels can encapsulate a same- layer or higher layer protocol and transport the result over IP through a tunnel that is created between two routers.

# Licensing Requirements for Layer 3 Interfaces

This feature does not require a license. Any feature not included in a license package is bundled with the Cisco NX-OS system images and is provided at no extra charge to you. For a complete explanation of the Cisco NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*.

# Guidelines and Limitations for Layer 3 Interfaces

Layer 3 interfaces have the following configuration guidelines and limitations:

- If you change a Layer 3 interface to a Layer 2 interface, Cisco NX-OS shuts down the interface, reenables the interface, and removes all configuration specific to Layer 3.

- If you change a Layer 2 interface to a Layer 3 interface, Cisco NX-OS shuts down the interface, reenables the interface, and deletes all configuration specific to Layer 2.
- 
- 
- Cisco Nexus 3016 will punt multicast Layer 2 traffic to the CPU if the Layer 3 MTU is not the same for all Layer 3 interfaces, and if the MTU QoS was changed to jumbo. All Layer 3 interfaces must have the same Layer 3 MTU to avoid this issue.

# Default Settings for Layer 3 Interfaces

The default setting for the Layer 3 Admin state is Shut.

# SVI Autostate Disable

The SVI Autostate Disable feature enables the Switch Virtual Interface (SVI) to be in the “up” state even if no interface is in the “up” state in the corresponding VLAN.

An SVI is also a virtual routed interface that connects a VLAN on the device to the Layer 3 router engine on the same device. The ports in a VLAN determine the operational state of the corresponding SVI. An SVI interface on a VLAN comes “up” when at least one port in the corresponding VLAN is in the Spanning Tree Protocol (STP) forwarding state. Similarly, the SVI interface goes “down” when the last STP forwarding port goes down or to any other state. This characteristic of SVI is called 'Autostate'.

You can create SVIs to define Layer 2 or Layer 3 boundaries on VLANs, or use the SVI interface to manage devices. In the second scenario, the SVI Autostate Disable feature ensures that the SVI interface is in the “up” state even if no interface is in the “up” state in the corresponding VLAN.

# DHCP Client Discovery

Cisco NX-OS Release 6.0(2)U3(1) introduced DHCP client discovery on SVIs. Cisco NX-OS Release 6.0(2)U4(1) adds DHCP client discovery support for IPv6 addresses and physical Ethernet and management interfaces. You can configure the IP address of a DHCP client by using the **ip address dhcp** or **ipv6 address dhcp** command. These commands sends a request from the DHCP client to the DHCP server soliciting an IPv4 or IPv6 address from the DHCP server. The DHCP client on the Cisco Nexus switch identifies itself to the DHCP server. The DHCP server uses this identifier to send the IP address back to the DHCP client.

When a DHCP client is configured on the SVI with the DHCP server sending router and DNS options, the **ip route** 0.0.0.0/0 *router-ip* and **ip name-server** *dns-ip* commands are configured on the switch automatically.

If the switch is reloaded and, at the same time, the router and DNS options are disabled on the server side, after the switch comes up, a new IP address is assigned to the SVI. However, the stale **ip route** command and **ip name-server** command will still exist in the switch configuration. You must manually remove these commands from the configuration.

## Limitations for Using DHCP Client Discovery on Interfaces

The following are the limitations for using DHCP client discovery on interfaces:

- This feature is supported only on physical Ethernet interfaces, management interfaces, and SVIs.
- Starting with Cisco NX-OS Release 6.0(2)U4(1), this feature is supported on non-default virtual routing and forwarding (VRF) instances as well.
- The DNS server and default router option-related configurations are saved in the startup configuration when you enter the **copy running-config startup-config** command. When you reload the switch, if this configuration is not applicable, you might have to remove it.
- You can configure a maximum of six DNS servers on the switch, which is a switch limitation. This maximum number includes the DNS servers configured by the DHCP client and the DNS servers configured manually.
- If the number of DNS servers configured on the switch is more than six, and if you get a DHCP offer for an SVI with DNS option set, the IP address is not assigned to the SVI.

# MAC-Embedded IPv6 Address

Beginning with Cisco NX-OS Release 6.0(2)U4(1), BGP allows an IPv4 prefix to be carried over an IPv6 next-hop. The IPv6 next-hop is leveraged to remove neighbor discovery (ND) related traffic from the network. To do this, the MAC address is embedded in the IPv6 address. Such an address is called a MAC Embedded IPv6 (MEv6) address. The router extracts the MAC address directly from the MEv6 address instead of going through ND. Local interface and next-hop MAC addresses are extracted from the IPv6 addresses.

On MEv6-enabled IPv6 interfaces, the same MEv6 extracted MAC address is used for IPv4 traffic as well. MEv6 is supported on all Layer 3 capable interfaces except SVIs.

**Important** When MEv6 is enabled on an interface, ping6 to the IPv6 link local address, OSPFv3, and BFDv6 are not supported on that interface.

# Configuring Layer 3 Interfaces

## Configuring a Routed Interface

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface ethernet** *slot/port* | Enters interface configuration mode. |
| **Step 3** | switch(conifg-if)# **no switchport** | Configures the interface as a Layer 3 interface and deletes any configuration specific to Layer 2 on this interface.<br>**Note** To convert a Layer 3 interface back into a Layer 2 interface, use the **switchport** command. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | switch(config-if)# [**ip**\|**ipv6**]*ip-address/length* | Configures an IP address for this interface. |
| **Step 5** | switch(config-if)# **medium** {**broadcast** \| **p2p**} | (Optional)<br>Configures the interface medium as either point to point or broadcast.<br>**Note** The default setting is broadcast, and this setting does not appear in any of the **show** commands. However, if you do change the setting to **p2p**, you will see this setting when you enter the **show running-config** command. |
| **Step 6** | switch(config-if)# **show interfaces** | (Optional)<br>Displays the Layer 3 interface statistics. |
| **Step 7** | switch(config-if)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to configure an IPv4-routed Layer 3 interface:

```
switch# configure terminal
switch(config)# interface ethernet 2/1
switch(config-if)# no switchport
switch(config-if)# ip address 192.0.2.1/8
switch(config-if)# copy running-config startup-config
```

# Configuring a Subinterface

### Before You Begin

- Configure the parent interface as a routed interface.
- Create the port-channel interface if you want to create a subinterface on that port channel.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch(config-if)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | switch(config)# **interface ethernet** *slot/port.number* | Enters interface configuration mode. The range for the *slot* is from 1 to 255. The range for the *port* is from 1 to 128. |
| **Step 3** | switch(config-if)# [**ip** \| **ipv6**] **address** *ip-address/length* | Configures an IP address for this interface. |
| **Step 4** | switch(config-if)# **encapsulation dot1Q** *vlan-id* | Configures IEEE 802.1Q VLAN encapsulation on the subinterface. The range for the *vlan-id* is from 2 to 4093. |
| **Step 5** | switch(config-if)# **show interfaces** | (Optional)<br>Displays the Layer 3 interface statistics. |
| **Step 6** | switch(config-if)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to create a subinterface:

```
switch# configure terminal
switch(config)# interface ethernet 2/1
switch(config-if)# ip address 192.0.2.1/8
switch(config-if)# encapsulation dot1Q 33
switch(config-if)# copy running-config startup-config
```

# Configuring the Bandwidth on an Interface

You can configure the bandwidth for a routed interface, port channel, or subinterface.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface ethernet** *slot/port* | Enters interface configuration mode. The range for the *slot* is from 1 to 255. The range for the *port* is from 1 to 128. |
| **Step 3** | switch(conifg-if)# **bandwidth** [*value* \| **inherit** [*value*]] | Configures the bandwidth parameter for a routed interface, port channel, or subinterface, as follows:<br>• *value*—Size of the bandwidth in kilobytes. The range is from 1 to 10000000.<br>• **inherit**—Indicates that all subinterfaces of this interface inherit either the bandwidth value (if a value is specified) or the bandwidth of the parent interface (if a value is not specified). |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | switch(config-if)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to configure Ethernet interface 2/1 with a bandwidth value of 80000:

```
switch# configure terminal
switch(config)# interface ethernet 2/1
switch(config-if)# bandwidth 80000
switch(config-if)# copy running-config startup-config
```

# Configuring a VLAN Interface

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **feature interface-vlan** | Enables VLAN interface mode. |
| **Step 3** | switch(config)# **interface vlan** *number* | Creates a VLAN interface. The *number* range is from 1 to 4094. |
| **Step 4** | switch(config-if)# [**ip** \| **ipv6** ] **address** *ip-address/length* | Configures an IP address for this interface. |
| **Step 5** | switch(config-if)# **no shutdown** | Brings the interface up administratively. |
| **Step 6** | switch(config-if)# **show interface vlan** *number* | (Optional)<br>Displays the VLAN interface statistics. The *number* range is from 1 to 4094. |
| **Step 7** | switch(config-if)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to create a VLAN interface:

```
switch# configure terminal
switch(config)# feature interface-vlan
switch(config)# interface vlan 10
switch(config-if)# ip address 192.0.2.1/8
switch(config-if)# copy running-config startup-config
```

# Configuring a Loopback Interface

### Before You Begin

Ensure that the IP address of the loopback interface is unique across all routers on the network.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface loopback** *instance* | Creates a loopback interface. The *instance* range is from 0 to 1023. |
| **Step 3** | switch(config-if)# [**ip** \| **ipv6** ] **address** *ip-address/length* | Configures an IP address for this interface. |
| **Step 4** | switch(config-if)# **show interface loopback** *instance* | (Optional)<br>Displays the loopback interface statistics. The *instance* range is from 0 to 1023. |
| **Step 5** | switch(config-if)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to create a loopback interface:

```
switch# configure terminal
switch(config)# interface loopback 0
switch(config-if)# ip address 192.0.2.100/8
switch(config-if)# copy running-config startup-config
```

# Assigning an Interface to a VRF

### Before You Begin

Assign the IP address for a tunnel interface after you have configured the interface for a VRF.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface** *interface-typenumber* | Enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | switch(conifg-if)#**vrf member** *vrf-name* | Adds this interface to a VRF. |
| **Step 4** | switch(config-if)# [**ip** \| **ipv6**]*ip-address/length* | Configures an IP address for this interface. You must do this step after you assign this interface to a VRF. |
| **Step 5** | switch(config-if)# **show vrf** [*vrf-name*] **interface** *interface-type number* | (Optional)<br>Displays VRF information. |
| **Step 6** | switch(config-if)# **show interfaces** | (Optional)<br>Displays the Layer 3 interface statistics. |
| **Step 7** | switch(config-if)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to add a Layer 3 interface to the VRF:

```
switch# configure terminal
switch(config)# interface loopback 0
switch(config-if)# vrf member RemoteOfficeVRF
switch(config-if)# ip address 209.0.2.1/16
switch(config-if)# copy running-config startup-config
```

# Configuring an Interface MAC Address

You can configure a static MAC address on SVI, Layer 3 interfaces, port channels, Layer 3 subinterfaces, and tunnel interfaces. You can also configure static MAC addresses on a range of ports and port channels. However, all ports must be in Layer 3. Even if one port in the range of ports is in Layer 2, the command is rejected and an error message appears.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface ethernet** *slot/port* | Enters interface configuration mode. |
| **Step 3** | switch(config-if)# [**no**] **mac-address** *static router MAC address* | Configures the interface MAC address. The **no** form removes the configuration. You can enter the MAC address in any one of the four supported formats:<br>• E.E.E<br>• EE-EE-EE-EE-EE-EE<br>• EE:EE:EE:EE:EE:EE |

| | Command or Action | Purpose |
|---|---|---|
| | | • EEEE.EEEE.EEEE<br>Do not enter any of the following invalid MAC addresses:<br>• Null MAC address—0000.0000.0000<br>• Broadcast MAC address—FFFF.FFFF.FFFF<br>• Multicast MAC address—0100.DAAA.ADDD |
| **Step 4** | switch(config-if)# **show interface ethernet** *slot/port* | (Optional) Displays all information for the interface. |

This example shows how to configure an interface MAC address:

```
switch# configure terminal
switch(config)# interface ethernet 3/3
switch(config-if)# mac-address aaaa.bbbb.dddd
switch(config-if)# show interface ethernet 3/3
switch(config-if)#
```

# Configuring a MAC-Embedded IPv6 Address

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface** *type slot/port* | Enters the interface configuration mode for the specified interface. |
| **Step 3** | switch(config-if)# **no switchport** | Configures the interface as a Layer 3 interface and deletes any configuration specific to Layer 2 on this interface.<br>**Note** To convert a Layer 3 interface back into a Layer 2 interface, use the **switchport** command. |
| **Step 4** | switch(config-if)# **mac-address ipv6-extract** | Extracts the MAC address embedded in the IPv6 address configured on the interface.<br>**Note** The MEv6 configuration is currently not supported with the EUI-64 format of IPv6 address. |
| **Step 5** | switch(config-if)# **ipv6 address** *ip-address/length* | Configures an IPv6 address for this interface. |
| **Step 6** | switch(config-if)# **ipv6 nd mac-extract** [**exclude nud-phase**] | Extracts the next-hop MAC address embedded in a next-hop IPv6 address.<br>The **exclude nud-phase** option blocks packets during the ND phase only. When the **exclude nud-phase** option is not |

| | Command or Action | Purpose |
|---|---|---|
| | | specified, packets are blocked during both ND and Neighbor Unreachability Detection (NUD) phases. |
| **Step 7** | switch(config)# **show ipv6 icmp interface** *type slot/port* | (Optional)<br>Displays IPv6 Internet Control Message Protocol version 6 (ICMPv6) interface information. |

This example shows how to configure a MAC-embedded IPv6 address with ND mac-extract enabled:

```
switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)# interface ethernet 1/3
switch(config-if)# no switchport
switch(config-if)# mac-address ipv6-extract
switch(config-if)# ipv6 address 2002:1::10/64
switch(config-if)# ipv6 nd mac-extract
switch(config-if)# show ipv6 icmp interface ethernet 1/3
ICMPv6 Interfaces for VRF "default"
Ethernet1/3, Interface status: protocol-up/link-up/admin-up
  IPv6 address: 2002:1::10
  IPv6 subnet:  2002:1::/64
  IPv6 interface DAD state:  VALID
  ND mac-extract : Enabled
  ICMPv6 active timers:
      Last Neighbor-Solicitation sent: 00:01:39
      Last Neighbor-Advertisement sent: 00:01:40
      Last Router-Advertisement sent: 00:01:41
      Next Router-Advertisement sent in: 00:03:34
  Router-Advertisement parameters:
      Periodic interval: 200 to 600 seconds
      Send "Managed Address Configuration" flag: false
      Send "Other Stateful Configuration" flag: false
      Send "Current Hop Limit" field: 64
      Send "MTU" option value: 1500
      Send "Router Lifetime" field: 1800 secs
      Send "Reachable Time" field: 0 ms
      Send "Retrans Timer" field: 0 ms
      Suppress RA: Disabled
      Suppress MTU in RA: Disabled
  Neighbor-Solicitation parameters:
      NS retransmit interval: 1000 ms
  ICMPv6 error message parameters:
      Send redirects: true
      Send unreachables: false
  ICMPv6-nd Statisitcs (sent/received):
      RAs: 3/0, RSs: 0/0, NAs: 2/0, NSs: 7/0, RDs: 0/0
      Interface statistics last reset: never
switch(config)#
```

This example shows how to configure a MAC-embedded IPv6 address with ND mac-extract (excluding NUD phase) enabled:

```
switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)# interface ethernet 1/5
switch(config-if)# no switchport
switch(config-if)# mac-address ipv6-extract
switch(config-if)# ipv6 address 2002:2::10/64
switch(config-if)# ipv6 nd mac-extract exclude nud-phase
switch(config-if)# show ipv6 icmp interface ethernet 1/5
ICMPv6 Interfaces for VRF "default"
Ethernet1/5, Interface status: protocol-up/link-up/admin-up
```

```
  IPv6 address: 2002:2::10
  IPv6 subnet:  2002:2::/64
  IPv6 interface DAD state:  VALID
  ND mac-extract : Enabled (Excluding NUD Phase)
  ICMPv6 active timers:
      Last Neighbor-Solicitation sent: 00:06:45
      Last Neighbor-Advertisement sent: 00:06:46
      Last Router-Advertisement sent: 00:02:18
      Next Router-Advertisement sent in: 00:02:24
  Router-Advertisement parameters:
      Periodic interval: 200 to 600 seconds
      Send "Managed Address Configuration" flag: false
      Send "Other Stateful Configuration" flag: false
      Send "Current Hop Limit" field: 64
      Send "MTU" option value: 1500
      Send "Router Lifetime" field: 1800 secs
      Send "Reachable Time" field: 0 ms
      Send "Retrans Timer" field: 0 ms
      Suppress RA: Disabled
      Suppress MTU in RA: Disabled
  Neighbor-Solicitation parameters:
      NS retransmit interval: 1000 ms
  ICMPv6 error message parameters:
      Send redirects: true
      Send unreachables: false
  ICMPv6-nd Statisitcs (sent/received):
      RAs: 6/0, RSs: 0/0, NAs: 2/0, NSs: 7/0, RDs: 0/0
      Interface statistics last reset: never
switch(config-if)#
```

# Configuring SVI Autostate Disable

You can configure a SVI to remain active even if no interfaces are up in the corresponding VLAN. This enhancement is called Autostate Disable.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **system default interface-vlan autostate** | Reenables the system default autostate behavior on Switching Virtual Interface (SVI) in a VLAN. Use the no form of the command to disable the autostate behavior on SVI. |
| **Step 3** | switch(config)# **feature interface-vlan** | Enables the creation of VLAN interfaces SVI. |
| **Step 4** | switch(config)# **interface vlan** *vlan id* | Disables the VLAN interface and enters interface configuration mode. |
| **Step 5** | (config-if)# [**no**] **autostate** | Disables the default autostate behavior of SVIs on the VLAN interface. |
| **Step 6** | (config-if)# **end** | Returns to privileged EXEC mode. |
| **Step 7** | **show running-config interface vlan** *vlan id* | (Optional) Displays the running configuration for a specific port channel. |

This example shows how to configure the SVI Autostate Disable feature:

```
switch# configure terminal
switch(config)# system default interface-vlan autostate
switch(config)# feature interface-vlan
switch(config)# interface vlan 2
switch(config-if)# no autostate
switch(config-if)# end
```

# Configuring a DHCP Client on an Interface

You can configure the IP address of a DHCP client on an SVI, a management interface, or a physical Ethernet interface.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface ethernet** *type slot/port* \| **mgmt** *mgmt-interface-number* \| **vlan** *vlan id* | Creates a physical Ethernet interface, a management interface, or a VLAN interface.<br>The range of *vlan id* is from 1 to 4094. |
| **Step 3** | switch(config-if)# [**no**] **ip** \| **ipv6 address dhcp** | Requests the DHCP server for an IPv4 or IPv6 address.<br>The **no** form of this command removes any address that was acquired. |
| **Step 4** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to configure the IP address of a DHCP client on an SVI:

```
switch# configure terminal
switch(config)# interface vlan 15
switch(config-if)# ip address dhcp
```

This example shows how to configure an IPv6 address of a DHCP client on a management interface:

```
switch# configure terminal
switch(config)# interface mgmt 0
switch(config-if)# ipv6 address dhcp
```

# Verifying the Layer 3 Interfaces Configuration

Use one of the following commands to verify the configuration:

| Command | Purpose |
|---|---|
| **show interface ethernet** *slot/port* | Displays the Layer 3 interface configuration, status, and counters (including the 5-minute exponentially decayed moving average of inbound and outbound packet and byte rates). |
| **show interface ethernet** *slot/port* **brief** | Displays the Layer 3 interface operational status. |
| **show interface ethernet** *slot/port* **capabilities** | Displays the Layer 3 interface capabilities, including port type, speed, and duplex. |
| **show interface ethernet** *slot/port* **description** | Displays the Layer 3 interface description. |
| **show interface ethernet** *slot/port* **status** | Displays the Layer 3 interface administrative status, port mode, speed, and duplex. |
| **show interface ethernet** *slot/port***.***number* | Displays the subinterface configuration, status, and counters (including the f-minute exponentially decayed moving average of inbound and outbound packet and byte rates). |
| **show interface port-channel** *channel-id***.***number* | Displays the port-channel subinterface configuration, status, and counters (including the 5-minute exponentially decayed moving average of inbound and outbound packet and byte rates). |
| **show interface loopback** *number* | Displays the loopback interface configuration, status, and counters. |
| **show interface loopback** *number* **brief** | Displays the loopback interface operational status. |
| **show interface loopback** *number* **description** | Displays the loopback interface description. |
| **show interface loopback** *number* **status** | Displays the loopback interface administrative status and protocol status. |
| **show interface vlan** *number* | Displays the VLAN interface configuration, status, and counters. |
| **show interface vlan** *number* **brief** | Displays the VLAN interface operational status. |
| **show interface vlan** *number* **description** | Displays the VLAN interface description. |
| **show interface vlan** *number* **private-vlan mapping** | Displays the VLAN interface private VLAN information. |
| **show interface vlan** *number* **status** | Displays the VLAN interface administrative status and protocol status. |

# Triggering the Layer 3 Interface Consistency Checker

You can manually trigger the Layer 3 interface consistency checker to compare the hardware and software configuration of all physical interfaces in a module and display the results. To manually trigger the Layer 3 Interface consistency checker and display the results, use the following command in any mode:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **show consistency-checker l3-interface module** *slot* | Starts the Layer 3 interface consistency check on all Layer 3 physical interfaces of a module that are up and displays its results. |

This example shows how to trigger the Layer 3 interface consistency check and display its results:

```
switch# show consistency-checker l3-interface module 1
L3 LIF Checks: L3 Vlan, CML Flags, IPv4 Enable
Consistency Check: PASSED
No inconsistencies found for:
   Ethernet1/17
   Ethernet1/49
   Ethernet1/50
```

# Monitoring Layer 3 Interfaces

Use one of the following commands to display statistics about the feature:

| Command | Purpose |
|---|---|
| **load-interval** *seconds* \| **counter** {**1** \| **2** \| **3**} *seconds* | Sets three different sampling intervals to bit-rate and packet-rate statistics. The range is from 5 seconds to 300 seconds. |
| **show interface ethernet** *slot*/*port* **counters** | Displays the Layer 3 interface statistics (unicast, multicast, and broadcast). |
| **show interface ethernet** *slot*/*port* **counters brief** *load-interval-id* | Displays the Layer 3 interface input and output counters.<br>The load interval ID specifies a single load interval ID to display the input and output rates.<br>The load interval ID ranges between 1 and 3. |
| **show interface ethernet** *slot*/*port* **counters detailed** [**all**] | Displays the Layer 3 interface statistics. You can optionally include all 32-bit and 64-bit packet and byte counters (including errors). |
| **show interface ethernet** *slot*/*port* **counters error** | Displays the Layer 3 interface input and output errors. |

| Command | Purpose |
|---|---|
| **show interface ethernet** *slot*/*port* **counters snmp** | Displays the Layer 3 interface counters reported by SNMP MIBs. You cannot clear these counters. |
| **show interface ethernet** *slot*/*port*.*number* **counters** | Displays the subinterface statistics (unicast, multicast, and broadcast). |
| **show interface port-channel** *channel-id*.*number* **counters** | Displays the port-channel subinterface statistics (unicast, multicast, and broadcast). |
| **show interface loopback** *number* **counters** | Displays the loopback interface input and output counters (unicast, multicast, and broadcast). |
| **show interface loopback** *number* **counters detailed** [**all**] | Displays the loopback interface statistics. You can optionally include all 32-bit and 64-bit packet and byte counters (including errors). |
| **show interface loopback** *number* **counters errors** | Displays the loopback interface input and output errors. |
| **show interface vlan** *number* **counters** | Displays the VLAN interface input and output counters (unicast, multicast, and broadcast). |
| **show interface vlan** *number* **counters detailed** [*all*] | Displays the VLAN interface statistics. You can optionally include all Layer 3 packet and byte counters (unicast and multicast). |
| **show interface vlan** *counters* **snmp** | Displays the VLAN interface counters reported by SNMP MIBs. You cannot clear these counters. |

# Configuration Examples for Layer 3 Interfaces

This example shows how to configure Ethernet subinterfaces:

```
switch# configuration terminal
switch(config)# interface ethernet 2/1.10
switch(config-if)# description Layer 3 for VLAN 10
switch(config-if)# encapsulation dot1q 10
switch(config-if)# ip address 192.0.2.1/8
switch(config-if)# copy running-config startup-config
```

This example shows how to configure a VLAN interface:

```
switch# configuration terminal
switch(config)# interface vlan 100
switch(config-if)# no switchport

switch(config-if)# ipv6 address 33:0DB::2/8
switch(config-if)# copy running-config startup-config
```

This example shows how to configure Switching Virtual Interface (SVI) Autostate Disable:

```
switch# configure terminal
switch(config)# system default interface-vlan autostate
switch(config)# feature interface-vlan
switch(config)# interface vlan 2
switch(config-if)# no autostate
switch(config-if)# end
switch# show running-config interface vlan 2
```

This example shows how to configure a loopback interface:

```
switch# configuration terminal
switch(config)# interface loopback 3
switch(config-if)# no switchport
switch(config-if)# ip address 192.0.2.2/32
switch(config-if)# copy running-config startup-config
```

This example shows how to configure the three sample load intervals for an Ethernet port:

```
switch# configure terminal
switch(config)# interface ethernet 1/3
switch(config-if)# load-interval counter 1 5
switch(config-if)# load-interval counter 2 135
switch(config-if)# load-interval counter 3 225
switch(config-if)#
```

# Related Documents for Layer 3 Interfaces

| Related Topics | Document Title |
|---|---|
| Command syntax | *Cisco Nexus 3000 Series Command Reference* |
| IP | "Configuring IP" chapter in the *Cisco Nexus 3000 Series NX-OS Unicast Routing Configuration Guide* |
| VLAN | "Configuring VLANs" chapter in the *Cisco Nexus 3000 Series NX-OS Layer 2 Switching Configuration Guide* |

# MIBs for Layer 3 Interfaces

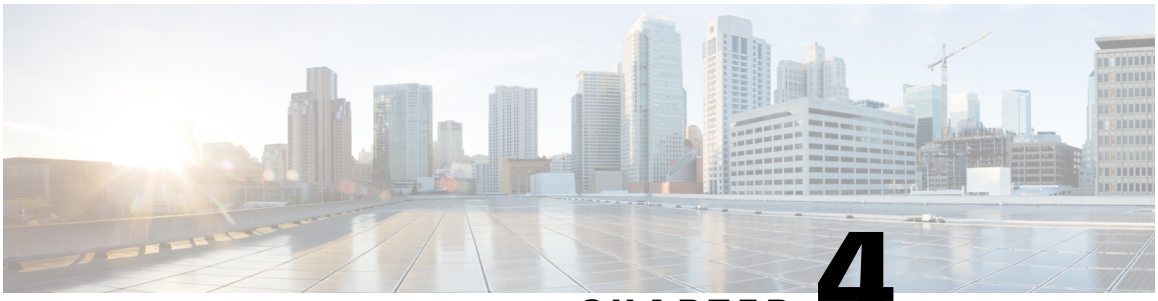
| MIB | MIB Link |
|---|---|
| CISCO-IF-EXTENSION-MIB | To locate and download MIBs, go to the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml |
| ETHERLIKE-MIB | |

# Standards for Layer 3 Interfaces

No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.

# Feature History for Layer 3 Interfaces

| Feature Name | Release | Feature Information |
|---|---|---|
| **show interface vlan** *vlan-id* **counters** command | 5.0(3)U3(1) | The **show interface vlan** *vlan-id* **counters** command has been enhanced to correctly show input and output packet counts. |



CHAPTER 4

# Configuring Port Channels

This chapter contains the following sections:



## Information About Port Channels

A port channel bundles individual interfaces into a group to provide increased bandwidth and redundancy. Port channeling also load balances traffic across these physical interfaces. The port channel stays operational as long as at least one physical interface within the port channel is operational.

You create a port channel by bundling compatible interfaces. You can configure and run either static port channels or port channels running the Link Aggregation Control Protocol (LACP).

Any configuration changes that you apply to the port channel are applied to each member interface of that port channel. For example, if you configure Spanning Tree Protocol (STP) parameters on the port channel, Cisco NX-OS applies those parameters to each interface in the port channel.

You can use static port channels, with no associated protocol, for a simplified configuration. For more efficient use of the port channel, you can use the Link Aggregation Control Protocol (LACP), which is defined in IEEE 802.3ad. When you use LACP, the link passes protocol packets.

### Related Topics


LACP Overview, on page 61

# Understanding Port Channels

Using port channels, Cisco NX-OS provides wider bandwidth, redundancy, and load balancing across the channels.

You can collect ports into a static port channel or you can enable the Link Aggregation Control Protocol (LACP). Configuring port channels with LACP requires slightly different steps than configuring static port channels. For information on port channel configuration limits, see the *Verified Scalability* document for your platform. For more information about load balancing, see Load Balancing Using Port Channels, on page 58.

**Note** Cisco NX-OS does not support Port Aggregation Protocol (PAgP) for port channels.

A port channel bundles individual links into a channel group to create a single logical link that provides the aggregate bandwidth of several physical links. If a member port within a port channel fails, traffic previously carried over the failed link switches to the remaining member ports within the port channel.

Each port can be in only one port channel. All the ports in a port channel must be compatible; they must use the same speed and operate in full-duplex mode. When you are running static port channels without LACP, the individual links are all in the on channel mode; you cannot change this mode without enabling LACP.

**Note** You cannot change the mode from ON to Active or from ON to Passive.

You can create a port channel directly by creating the port-channel interface, or you can create a channel group that acts to aggregate individual ports into a bundle. When you associate an interface with a channel group, Cisco NX-OS creates a matching port channel automatically if the port channel does not already exist. You can also create the port channel first. In this instance, Cisco NX-OS creates an empty channel group with the same channel number as the port channel and takes the default configuration.

**Note** A port channel is operationally up when at least one of the member ports is up and that port's status is channeling. The port channel is operationally down when all member ports are operationally down.

# Compatibility Requirements

When you add an interface to a port channel group, Cisco NX-OS checks certain interface attributes to ensure that the interface is compatible with the channel group. Cisco NX-OS also checks a number of operational attributes for an interface before allowing that interface to participate in the port-channel aggregation.

The compatibility check includes the following operational attributes:

- Port mode
- Access VLAN
- Trunk native VLAN
- Allowed VLAN list
- Speed

- 802.3x flow control setting
- MTU
- Broadcast/Unicast/Multicast Storm Control setting
- Priority-Flow-Control
- Untagged CoS

Use the **show port-channel compatibility-parameters** command to see the full list of compatibility checks that Cisco NX-OS uses.

You can only add interfaces configured with the channel mode set to on to static port channels. You can also only add interfaces configured with the channel mode as active or passive to port channels that are running LACP. You can configure these attributes on an individual member port.

When the interface joins a port channel, the following individual parameters are replaced with the values on the port channel:

- Bandwidth
- MAC address
- Spanning Tree Protocol

The following interface parameters remain unaffected when the interface joins a port channel:

- Description
- CDP
- LACP port priority
- Debounce

After you enable forcing a port to be added to a channel group by entering the **channel-group force** command, the following two conditions occur:

- When an interface joins a port channel, the following parameters are removed and they are operationally replaced with the values on the port channel; however, this change will not be reflected in the running configuration for the interface:
  - QoS
  - Bandwidth
  - Delay
  - STP
  - Service policy
  - ACLs
- When an interface joins or leaves a port channel, the following parameters remain unaffected:
  - Beacon
  - Description
  - CDP

- LACP port priority
- Debounce
- UDLD
- Shutdown
- SNMP traps

# Load Balancing Using Port Channels

Cisco NX-OS load balances traffic across all operational interfaces in a port channel by reducing part of the binary pattern formed from the addresses in the frame to a numerical value that selects one of the links in the channel. Port channels provide load balancing by default.

The basic configuration uses the following criteria to select the link:

- For a Layer 2 frame, it uses the source and destination MAC addresses.
- For a Layer 3 frame, it uses the source and destination MAC addresses and the source and destination IP addresses.
- For a Layer 4 frame, it uses the source and destination MAC addresses and the source and destination IP addresses.

**Note** You have the option to include the source and destination port number for the Layer 4 frame.

You can configure the switch to use one of the following methods (see the following table for more details) to load balance across the port channel:

- Destination MAC address
- Source MAC address
- Source and destination MAC address
- Destination IP address
- Source IP address
- Source and destination IP address
- Destination TCP/UDP port number
- Source TCP/UDP port number
- Source and destination TCP/UDP port number

*Table 3: Port Channel Load-Balancing Criteria*

| Configuration | Layer 2 Criteria | Layer 3 Criteria | Layer 4 Criteria |
|---|---|---|---|
| Destination MAC | Destination MAC | Destination MAC | Destination MAC |
| Source MAC | Source MAC | Source MAC | Source MAC |
| Source and destination MAC | Source and destination MAC | Source and destination MAC | Source and destination MAC |
| Destination IP | Destination MAC | Destination MAC, destination IP | Destination MAC, destination IP |
| Source IP | Source MAC | Source MAC, source IP | Source MAC, source IP |
| Source and destination IP | Source and destination MAC | Source and destination MAC, source and destination IP | Source and destination MAC, source and destination IP |
| Destination TCP/UDP port | Destination MAC | Destination MAC, destination IP | Destination MAC, destination IP, destination port |
| Source TCP/UDP port | Source MAC | Source MAC, source IP | Source MAC, source IP, source port |
| Source and destination TCP/UDP port | Source and destination MAC | Source and destination MAC, source and destination IP | Source and destination MAC, source and destination IP, source and destination port |

Use the option that provides the balance criteria with the greatest variety in your configuration. For example, if the traffic on a port channel is going only to a single MAC address and you use the destination MAC address as the basis of port-channel load balancing, the port channel always chooses the same link in that port channel; using source addresses or IP addresses might result in better load balancing.

The unicast and multicast traffic is load-balanced across port-channel links based on configured load-balancing algorithm shown in **show port-channel load-balancing** command output. The multicast traffic uses the following methods for load balancing with port channels:

- Multicast traffic with Layer 4 information - Source IP address, source port, destination IP address, destination port
- Multicast traffic without Layer 4 information - Source IP address, destination IP address
- Non-IP multicast traffic - Source MAC address, destination MAC address

**Note** The hardware multicast hw-hash command is not supported on Cisco Nexus 3000 Series switches and Cisco Nexus 3100 Series switches. It is recommended not to configure this command on these switches. By default, Cisco Nexus 3000 Series switches and Cisco Nexus 3100 Series switches hash multicast traffic.

**Note** The hardware multicast hw-hash command is not supported on Cisco Nexus 3500 Series switches. It is recommended not to configure this command on these switches.

# Resilient Hashing

With the exponential increase in the number of physical links used in data centers, there is also the potential for an increase in the number of failed physical links. In static hashing systems that are used for load balancing flows across members of port channels or Equal Cost Multipath (ECMP) groups, each flow is hashed to a link. If a link fails, all flows are rehashed across the remaining working links. This rehashing of flows to links results in some packets being delivered out of order even for those flows that were not hashed to the failed link.

This rehashing also occurs when a link is added to the port channel or Equal Cost Multipath (ECMP) group. All flows are rehashed across the new number of links, which results in some packets being delivered out of order. Resilient hashing supports only unicast traffic.

The resilient hashing system in Cisco Nexus 3100 Series switches maps flows to physical ports. In case a link fails, the flows assigned to the failed link are redistributed uniformly among the working links. The existing flows through the working links are not rehashed and their packets are not delivered out of order.

Resilient hashing is supported only by ECMP groups and on port channel interfaces. When a link is added to the port channel or ECMP group, some of the flows hashed to the existing links are rehashed to the new link, but not across all existing links.

Resilient hashing supports IPv4 and IPv6 unicast traffic, but it does not support IPv4 multicast traffic.

# Hashing for NVGRE Traffic

You can use Network Virtualization using Generic Routing Encapsulatoin (NVGRE) to virtualize and extend a network so that Layer 2 and Layer 3 topologies are created across distributed data centers. NVGRE uses encapsulation and tunneling. NVGRE endpoints are network devices that act as interfaces between the physical and virtualized networks.

Data frames are encapsulated or decapsulated at NVGRE endpoints using GRE tunneling. The endpoints obtain the destination address for each data frame from the Tenant Network Identifier (TNI). The Key field in the GRE header holds the 24-bit TNI. Each TNI represents a specific tenant's subnet address.

Cisco NX-OS Release 6.0(2)U2(1) supports hashing for transit NVGRE traffic. You can configure the switch to include the GRE Key field present in the GRE header in hash computations when NVGRE traffic is forwarded over a port channel or an Equal Cost Multipath (ECMP).

# Symmetric Hashing

To be able to effectively monitor traffic on a port channel, it is essential that each interface connected to a port channel receives both forward and reverse traffic flows. Normally, there is no guarantee that the forward and reverse traffic flows will use the same physical interface. However, when you enable symmetric hashing on the port channel, bidirectional traffic is forced to use the same physical interface and each physical interface in the port channel is effectively mapped to a set of flows.

Cisco NX-OS Release 6.0(2)U2(3) introduces symmetric hashing. When symmetric hashing is enabled, the parameters used for hashing, such as the source and destination IP address, are normalized before they are entered into the hashing algorithm. This process ensures that when the parameters are reversed (the source on the forward traffic becomes the destination on the reverse traffic), the hash output is the same. Therefore, the same interface is chosen.

Symmetric hashing is supported only on Cisco Nexus 3100 Series switches.

Only the following load-balancing algorithms support symmetric hashing:

- source-dest-ip-only
- source-dest-port-only
- source-dest-ip
- source-dest-port
- source-dest-ip-gre

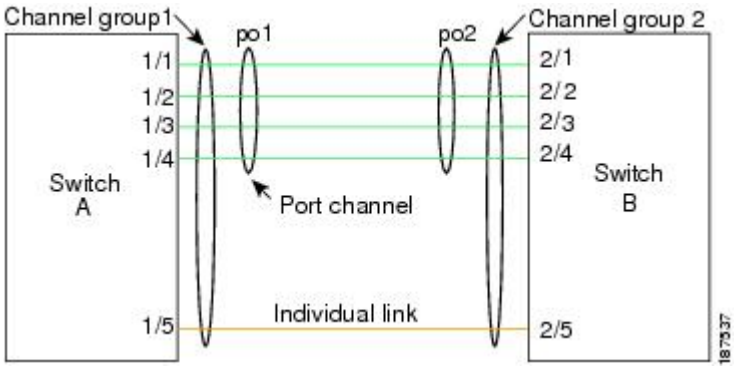
# Understanding LACP

## LACP Overview

**Note** You must enable the LACP feature before you can configure and use LACP functions.

The following figure shows how individual links can be combined into LACP port channels and channel groups as well as function as individual links.

***Figure 4: Individual Links Combined into a Port Channel***




With LACP, just like with static port channels, you can bundle up to 16 interfaces in a channel group.

**Note** When you delete the port channel, Cisco NX-OS automatically deletes the associated channel group. All member interfaces revert to their previous configuration.

You cannot disable LACP while any LACP configurations are present.

## LACP ID Parameters

LACP uses the following parameters:

- LACP system priority—Each system that runs LACP has an LACP system priority value. You can accept the default value of 32768 for this parameter, or you can configure a value between 1 and 65535. LACP uses the system priority with the MAC address to form the system ID and also uses the system priority during negotiation with other devices. A higher system priority value means a lower priority.

**Note** The LACP system ID is the combination of the LACP system priority value and the MAC address.

- LACP port priority—Each port configured to use LACP has an LACP port priority. You can accept the default value of 32768 for the LACP port priority, or you can configure a value between 1 and 65535. LACP uses the port priority with the port number to form the port identifier. LACP uses the port priority to decide which ports should be put in standby mode when there is a limitation that prevents all compatible ports from aggregating and which ports should be put into active mode. A higher port priority value means a lower priority for LACP. You can configure the port priority so that specified ports have a lower priority for LACP and are most likely to be chosen as active links, rather than hot-standby links.
- LACP administrative key—LACP automatically configures an administrative key value equal to the channel-group number on each port configured to use LACP. The administrative key defines the ability of a port to aggregate with other ports. A port's ability to aggregate with other ports is determined by these factors:

- Port physical characteristics, such as the data rate, the duplex capability, and the point-to-point or shared medium state
- Configuration restrictions that you establish

## Channel Modes

Individual interfaces in port channels are configured with channel modes. When you run static port channels, with no protocol, the channel mode is always set to on. After you enable LACP globally on the device, you enable LACP for each channel by setting the channel mode for each interface to active or passive. You can configure either channel mode for individual links in the LACP channel group.

**Note** You must enable LACP globally before you can configure an interface in either the active or passive channel mode.

The following table describes the channel modes.

***Table 4: Channel Modes for Individual Links in a Port Channel***

| Channel Mode | Description |
|---|---|
| passive | LACP mode that places a port into a passive negotiating state, in which the port responds to LACP packets that it receives but does not initiate LACP negotiation. |
| active | LACP mode that places a port into an active negotiating state, in which the port initiates negotiations with other ports by sending LACP packets. |
| on | All static port channels, that is, that are not running LACP, remain in this mode. If you attempt to change the channel mode to active or passive before enabling LACP, the device returns an error message.<br>You enable LACP on each channel by configuring the interface in that channel for the channel mode as either active or passive. When an LACP attempts to negotiate with an interface in the on state, it does not receive any LACP packets and becomes an individual link with that interface; it does not join the LACP channel group. |

Both the passive and active modes allow LACP to negotiate between ports to determine if they can form a port channel, based on criteria such as the port speed and the trunking state. The passive mode is useful when you do not know whether the remote system, or partner, supports LACP.

Ports can form an LACP port channel when they are in different LACP modes as long as the modes are compatible as in the following examples:

- A port in active mode can form a port channel successfully with another port that is in active mode.
- A port in active mode can form a port channel with another port in passive mode.
- A port in passive mode cannot form a port channel with another port that is also in passive mode because neither port will initiate negotiation.
- A port in on mode is not running LACP.

## LACP Marker Responders

Using port channels, data traffic may be dynamically redistributed due to either a link failure or load balancing. LACP uses the Marker Protocol to ensure that frames are not duplicated or reordered because of this redistribution. Cisco NX-OS supports only Marker Responders.

## LACP-Enabled and Static Port Channel Differences

The following table provides a brief summary of major differences between port channels with LACP enabled and static port channels. For information about the maximum configuration limits, see the *Verified Scalability* document for your device.

***Table 5: Port Channels with LACP Enabled and Static Port Channels***

| Configurations | Port Channels with LACP Enabled | Static Port Channels |
|---|---|---|
| Protocol applied | Enable globally. | Not applicable. |
| Channel mode of links | Can be either:<br>• Active<br>• Passive | Can only be On. |

## LACP Port Channel MinLinks

A port channel aggregates similar ports to provide increased bandwidth in a single manageable interface. The MinLinks feature allows you to define the minimum number of interfaces from a LACP bundle that must fail before the port channel goes down.

The LACP port channel MinLinks feature does the following:

- Configures the minimum number of port channel interfaces that must be linked and bundled in the LACP port channel.
- Prevents a low-bandwidth LACP port channel from becoming active.
- Causes the LACP port channel to become inactive if only a few active members ports supply the required minimum bandwidth.

**Note** The MinLinks feature works only with LACP port channels. The device allows you to configure this feature in non-LACP port channels, but the feature is not operational.

# Configuring Port Channels

## Creating a Port Channel

You can create a port channel before creating a channel group. Cisco NX-OS automatically creates the associated channel group.

**Note** If you want LACP-based port channels, you need to enable LACP.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface port-channel** *channel-number* | Specifies the port-channel interface to configure, and enters the interface configuration mode. The range is from 1 to 4096. Cisco NX-OS automatically creates the channel group if it does not already exist. |
| **Step 3** | switch(config)# **no interface port-channel** *channel-number* | Removes the port channel and deletes the associated channel group. |

This example shows how to create a port channel:

```
switch# configure terminal
switch (config)# interface port-channel 1
```

## Adding a Port to a Port Channel

You can add a port to a new channel group or to a channel group that already contains ports. Cisco NX-OS creates the port channel associated with this channel group if the port channel does not already exist.

**Note** If you want LACP-based port channels, you need to enable LACP.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface** *type slot/port* | Specifies the interface that you want to add to a channel group and enters the interface configuration mode. |
| **Step 3** | switch(config-if)# **switchport mode trunk** | (Optional)<br>Configures the interface as a trunk port. |
| **Step 4** | switch(config-if)# **switchport trunk** {**allowed vlan** *vlan-id* \| **native vlan** *vlan-id*} | (Optional)<br>Configures necessary parameters for a trunk port. |
| **Step 5** | switch(config-if)# **channel-group** *channel-number* | Configures the port in a channel group and sets the mode. The channel-number range is from 1 to 4096. Cisco NX-OS creates the port channel associated with this channel group if the port channel does not already exist. This is called implicit port channel creation. |
| **Step 6** | switch(config-if)# **no channel-group** | (Optional)<br>Removes the port from the channel group. The port reverts to its original configuration. |

This example shows how to add an Ethernet interface 1/4 to channel group 1:

```
switch# configure terminal
switch (config)# interface ethernet 1/4
switch(config-if)# switchport mode trunk
switch(config-if)# channel-group 1
```

# Configuring Load Balancing Using Port Channels

You can configure the load-balancing algorithm for port channels that applies to the entire device.

**Note** If you want LACP-based port channels, you need to enable LACP.

**Note** For load-balancing FC traffic across SAN PO members in Nexus 5672UP-16G switch, the **port-channel load-balance ethernet** command is not needed. The load-balancing happens by default.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **port-channel load-balance ethernet** {[**destination-ip** \| **destination-ip-gre** \| **destination-mac** \| **destination-port** \| **source-dest-ip** \| **source-dest-ip-gre** \| **source-dest-mac** \| **source-dest-port** \| **source-ip** \| **source-ip-gre** \| **source-mac** \| **source-port**] **symmetric** \| **crc-poly**} | Specifies the load-balancing algorithm and hash for the device. The range depends on the device. The default is **source-dest-mac**.<br>**Note** The optional **destination-ip-gre**, **source-dest-ip-gre** and **source-ip-gre** keywords are used to include the NVGRE key in the hash computation. Inclusion of the NVGRE key is not enabled by default in the case of port channels. You must configure it explicitly by using these optional keywords.<br>The optional **symmetric** keyword is used to enable or disable symmetric hashing. Symmetric hashing forces bi-directional traffic to use the same physical interface. Only the following load-balancing algorithms support symmetric hashing:<br>• source-dest-ip-only<br>• source-dest-port-only<br>• source-dest-ip<br>• source-dest-port<br>• source-dest-ip-gre |
| **Step 3** | switch(config)# **no port-channel load-balance ethernet** | (Optional)<br>Restores the default load-balancing algorithm of source-dest-mac. |
| **Step 4** | switch# **show port-channel load-balance** | (Optional)<br>Displays the port-channel load-balancing algorithm. |

This example shows how to configure source IP load balancing for port channels:

```
switch# configure terminal
switch (config)# port-channel load-balance ethernet source-ip
```

This example shows how to configure symmetric hashing for port channels:

```
switch# configure terminal
switch (config)# port-channel load-balance ethernet source-dest-ip-only symmetric
```

# Enabling LACP

LACP is disabled by default; you must enable LACP before you begin LACP configuration. You cannot disable LACP while any LACP configuration is present.

LACP learns the capabilities of LAN port groups dynamically and informs the other LAN ports. Once LACP identifies correctly matched Ethernet links, it facilitates grouping the links into an port channel. The port channel is then added to the spanning tree as a single bridge port.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **feature lacp** | Enables LACP on the switch. |
| **Step 3** | switch(config)# **show feature** | (Optional)<br>Displays enabled features. |

This example shows how to enable LACP:

```
switch# configure terminal
switch(config)# feature lacp
```

# Configuring the Channel Mode for a Port

You can configure the channel mode for each individual link in the LACP port channel as active or passive. This channel configuration mode allows the link to operate with LACP.

When you configure port channels with no associated protocol, all interfaces on both sides of the link remain in the on channel mode.

### Before You Begin

Ensure that you have enabled the LACP feature.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface** *type slot*/*port* | Specifies the interface to configure, and enters the interface configuration mode. |
| **Step 3** | switch(config-if)# **channel-group** *channel-number* [**force**] [**mode** {**on** \| **active** \| **passive**}] | Specifies the port mode for the link in a port channel. After LACP is enabled, you configure each link or the entire channel as active or passive.<br>**force—**Specifies that the LAN port be forcefully added to the channel group.<br>**mode—**Specifies the port channel mode of the interface.<br>**active—**Specifies that when you enable LACP, this command enables LACP on the specified interface. The interface is in an |

| | Command or Action | Purpose |
|---|---|---|
| | | active negotiating state in which the port initiates negotiations with other ports by sending LACP packets.<br>**on—**(Default mode) Specifies that all port channels that are not running LACP remain in this mode.<br>**passive—**Enables LACP only if an LACP device is detected. The interface is in a passive negotiation state in which the port responds to LACP packets that it receives but does not initiate LACP negotiation.<br>When you run port channels with no associated protocol, the channel mode is always on. |
| **Step 4** | switch(config-if)# **no channel-group** *number* **mode** | Returns the port mode to on for the specified interface. |

This example shows how to set the LACP-enabled interface to active port-channel mode for Ethernet interface 1/4 in channel group 5:

```
switch# configure terminal
switch (config)# interface ethernet 1/4
switch(config-if)# channel-group 5 mode active
```

This example shows how to forcefully add an interface to the channel group 5:

```
switch(config)# interface ethernet 1/1
switch(config-if)# channel-group 5 force
switch(config-if)#
```

# Configuring LACP Port Channel MinLinks

The MinLink feature works only with LACP port channels. The device allows you to configure this feature in non-LACP port channels, but the feature is not operational.

**Important** We recommend that you configure the LACP MinLink feature on both ends of your LACP port channel, that is, on both the switches. Configuring the **lacp min-links** command on only one end of the port channel might result in link flapping.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface port-channel** *number* | Specifies the interface to configure and enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | switch(config-if)# [**no**] **lacp min-links** *number* | Specifies the port channel interface to configure the number of minimum links and enters the interface configuration mode.<br>The default value for *number* is 1. The range is from 1 to 16.<br>Use the **no** form of this command to disable this feature. |
| **Step 4** | switch(config)# **show running-config interface port-channel** *number* | (Optional)<br>Displays the port channel MinLinks configuration. |

This example shows how to configure the minimum number of port channel interfaces on module 3:

```
switch# configure terminal
switch(config) # interface port-channel 3
switch(config-if) # lacp min-links 3
switch(config-if) #
```

# Configuring the LACP Fast Timer Rate

You can change the LACP timer rate to modify the duration of the LACP timeout. Use the **lacp rate** command to set the rate at which LACP control packets are sent to an LACP-supported interface. You can change the timeout rate from the default rate (30 seconds) to the fast rate (1 second). This command is supported only on LACP-enabled interfaces.

### Before You Begin

Ensure that you have enabled the LACP feature.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface** *type slot*/*port* | Specifies the interface to configure and enters the interface configuration mode. |
| **Step 3** | switch(config-if)# **lacp rate fast** | Configures the fast rate (one second) at which LACP control packets are sent to an LACP-supported interface. |

This example shows how to configure the LACP fast rate on Ethernet interface 1/4:

```
switch# configure terminal
switch(config)# interface ethernet 1/4
switch(config-if)# lacp rate fast
```

This example shows how to restore the LACP default rate (30 seconds) on Ethernet interface 1/4.

```
switch# configure terminal
switch(config)# interface ethernet 1/4
switch(config-if)# no lacp rate fast
```

# Configuring the LACP System Priority and System ID

The LACP system ID is the combination of the LACP system priority value and the MAC address.

### Before You Begin

Ensure that you have enabled the LACP feature.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **lacp system-priority** *priority* | Configures the system priority for use with LACP. Valid values are 1 through 65535, and higher numbers have lower priority. The default value is 32768. |
| **Step 3** | switch# **show lacp system-identifier** | (Optional)<br>Displays the LACP system identifier. |

This example shows how to set the LACP system priority to 2500:

```
switch# configure terminal
switch(config)# lacp system-priority 2500
```

# Configuring the LACP Port Priority

You can configure each link in the LACP port channel for the port priority.

### Before You Begin

Ensure that you have enabled the LACP feature.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface** *type slot*/*port* | Specifies the interface to configure, and enters the interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | switch(config-if)# **lacp port-priority** *priority* | Configures the port priority for use with LACP. Valid values are 1 through 65535, and higher numbers have lower priority. The default value is 32768. |

This example shows how to set the LACP port priority for Ethernet interface 1/4 to 40000:

```
switch# configure terminal
switch (config)# interface ethernet 1/4
switch(config-if)# lacp port priority 40000
```

# Verifying Port Channel Configuration

Use the following command to verify the port channel configuration information:

| Command | Purpose |
|---|---|
| **show interface port channel***chennal-number* | Displays the status of a port channel interface. |
| **show feature** | Displays enabled features. |
| **show resource** | Displays the number of resources currently available in the system. |
| **show lacp** {**counters** \| **interface** *type slot*/*port* \| **neighbor** \| **port-channel** \| **system-identifier**} | Displays LACP information. |
| **show port-channel compatibility-parameters** | Displays the parameters that must be the same among the member ports in order to join a port channel. |
| **show port-channel database** [**interface port-channel** *channel-number*] | Displays the aggregation state for one or more port-channel interfaces. |
| **show port-channel summary** | Displays a summary for the port channel interfaces. |
| **show port-channel traffic** | Displays the traffic statistics for port channels. |
| **show port-channel usage** | Displays the range of used and unused channel numbers. |
| **show port-channel database** | Displays information on current running of the port channel feature. |
| **show port-channel load-balance** | Displays information about load-balancing using port channels. |

# Triggering the Port Channel Membership Consistency Checker

You can manually trigger the port channel membership consistency checker to compare the hardware and software configuration of all ports in a port channel and display the results. To manually trigger the port channel membership consistency checker and display the results, use the following command in any mode:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **show consistency-checker membership port-channels** | Starts a port channel membership consistency check on the member ports of a port channel and displays its results. |

This example shows how to trigger a port channel membership consistency check and display its results:

```
switch# show consistency-checker membership port-channels
Checks: Trunk group and trunk membership table.
Consistency Check: PASSED
 No Inconsistencies found for port-channel1111:
  Module:1, Unit:0
     ['Ethernet1/4', 'Ethernet1/5', 'Ethernet1/6']
 No Inconsistencies found for port-channel2211:
  Module:1, Unit:0
     ['Ethernet1/7', 'Ethernet1/8', 'Ethernet1/9', 'Ethernet1/10']
 No Inconsistencies found for port-channel3311:
  Module:1, Unit:0
     ['Ethernet1/11', 'Ethernet1/12', 'Ethernet1/13', 'Ethernet1/14']
 No Inconsistencies found for port-channel4095:
  Module:1, Unit:0
     ['Ethernet1/33', 'Ethernet1/34', 'Ethernet1/35', 'Ethernet1/36', 'Ethernet1
/37', 'Ethernet1/38', 'Ethernet1/39', 'Ethernet1/40', 'Ethernet1/41', 'Ethernet1
/42', 'Ethernet1/43', 'Ethernet1/44', 'Ethernet1/45', 'Ethernet1/46', 'Ethernet1
/47', 'Ethernet1/48', 'Ethernet1/29', 'Ethernet1/30', 'Ethernet1/31', 'Ethernet1
/32']
```

# Verifying the Load-Balancing Outgoing Port ID

### Command Guidelines

The **show port-channel load-balance** command allows you to verify which ports a given frame is hashed to on a port channel. You need to specify the VLAN and the destination MAC in order to get accurate results.

**Note** Certain traffic flows are not subject to hashing such as when there is a single port in a port-channel.

The **show port-channel load-balance** command supports only unicast traffic hashing. Multicast traffic hashing is not supported.

To display the load-balancing outgoing port ID, perform one of the tasks:

| Command | Purpose |
|---|---|
| switch# **show port-channel load-balance forwarding-path interface port-channel** *port-channel-id* **vlan** vlan-id **dst-ip src-ip dst-mac src-mac l4-src-port** *port-id* **l4-dst-port** *port-id* **ether-type** *ether-type* **ip-proto** *ip-proto* | Displays the outgoing port ID. |

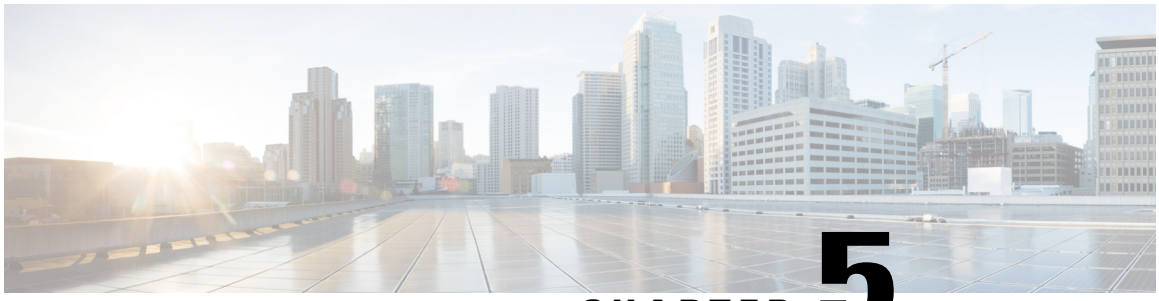
### Example

This example shows how to display the load balancing outgoing port ID:

```
switch# show port-channel load-balance forwarding-path interface port-channel 10 vlan 1
dst-ip 1.225.225.225 src-ip 1.1.10.10 src-mac aa:bb:cc:dd:ee:ff
l4-src-port 0 l4-dst-port 1
Missing params will be substituted by 0's. Load-balance Algorithm on switch: source-dest-port
 crc8_hash:204 Outgoing port id: Ethernet 1/1 Param(s) used to calculate load balance:
dst-port: 0
src-port: 0
dst-ip: 1.225.225.225
src-ip: 1.1.10.10
dst-mac: 0000.0000.0000
src-mac: aabb.ccdd.eeff
```

# Feature History for Port Channels

| Feature Name | Release | Feature Information |
|---|---|---|
| Minimum Links | 5.0(3)U3(1) | Added information about setting up and using the Minimum Links feature. |



CHAPTER 5

# Configuring IP Tunnels

This chapter contains the following sections:



## Information About IP Tunnels

IP tunnels can encapsulate a same-layer or higher-layer protocol and transport the result over IP through a tunnel created between two devices.

IP tunnels consists of the following three main components:

- Passenger protocol—The protocol that needs to be encapsulated. IPv4 is an example of a passenger protocol.
- Carrier protocol—The protocol that is used to encapsulate the passenger protocol. Cisco NX-OS supports generic routing encapsulation (GRE), and IP-in-IP encapsulation and decapsulation as carrier protocols.
- Transport protocol—The protocol that is used to carry the encapsulated protocol. IPv4 is an example of a transport protocol.

An IP tunnel takes a passenger protocol, such as IPv4, and encapsulates that protocol within a carrier protocol, such as GRE. The device then transmits this carrier protocol over a transport protocol, such as IPv4.

You configure a tunnel interface with matching characteristics on each end of the tunnel.

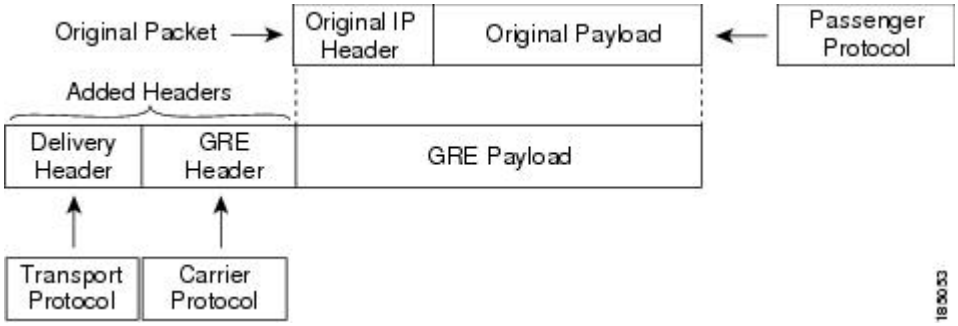
You must enable the tunnel feature before you can configure it.

# GRE Tunnels

You can use GRE as the carrier protocol for a variety of passenger protocols. The selection of tunnel interfaces can also be based on the PBR policy.

The figure shows the IP tunnel components for a GRE tunnel. The original passenger protocol packet becomes the GRE payload and the device adds a GRE header to the packet. The device then adds the transport protocol header to the packet and transmits it.

***Figure 5: GRE PDU***




# Point-to-Point IP-in-IP Tunnel Encapsulation and Decapsulation

Point-to-point IP-in-IP encapsulation and decapsulation is a type of tunnel that you can create to send encapsulated packets from a source tunnel interface to a destination tunnel interface. The selection of these tunnel interfaces can also be based on the PBR policy. This type of tunnel will carry both inbound and outbound traffic.

# Multi-Point IP-in-IP Tunnel Decapsulation

Multi-point IP-in-IP decapsulate-any is a type of tunnel that you can create to decapsulate packets from any number of IP-in-IP tunnels to one tunnel interface. This tunnel will not carry any outbound traffic. However, any number of remote tunnel endpoints can use a tunnel configured this way as their destination.

# Licensing Requirements for IP Tunnels

| Product | License Requirement |
|---|---|
| Cisco NX-OS | IP tunnels require an Enterprise Services license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*. |

# Prerequisites for IP Tunnels

IP tunnels have the following prerequisites:

- You must be familiar with TCP/IP fundamentals to configure IP tunnels.
- You are logged on to the switch.
- You have installed the Enterprise Services license for Cisco NX-OS.
- You must enable the tunneling feature in a device before you can configure and enable any IP tunnels.

# Guidelines and Limitations for IP Tunnels

IP tunnels have the following configuration guidelines and limitations:

- Cisco NX-OS software supports the GRE header defined in IETF RFC 2784. Cisco NX-OS software does not support tunnel keys and other options from IETF RFC 1701.
- The Cisco Nexus device supports the following maximum number tunnels:
  - GRE and IP-in-IP regular tunnels-8 tunnels
  - Multipoint IP-in-IP tunnels-32 tunnels
- Each tunnel will consume one Equal Cost Multipath (ECMP) adjacency.
- The Cisco Nexus device does not support the following features:
  - Path maximum transmission unit (MTU) discovery
  - Tunnel interface statistics
  - Access control lists (ACLs)
  - Unicast reverse path forwarding (URPF)
  - Multicast traffic and associated multicast protocols such as Internet Group Management Protocol (IGMP) and Protocol Independent Multicast (PIM)
- Cisco NX-OS software does not support the Web Cache Control Protocol (WCCP) on tunnel interfaces.

- Cisco NX-OS software supports only Layer-3 traffic.
- Cisco NX-OS software supports ECMP across tunnels and ECMP for tunnel destination.
- IPv6-in-IPv6 tunnels is not supported.
- Limited control protocols, such as Border Gateway Protocol (BGP), Open Shortest Path First (OSPF), and Enhanced Interior Gateway Routing Protocol (EIGRP), are supported for GRE tunnels.
- Starting with Release 6.0(2)U5(1), Cisco Nexus 3000 Series switches drop all the packets when the tunnel is not configured. The packets are also dropped when the tunnel is configured but the tunnel interface is not configured or the tunnel interface is in shut down state.

  Point to Point tunnel (Source and Destination) – Cisco Nexus 3000 Series switches decapsulate all IP-in-IP packets destined to it when the command **feature tunnel** is configured and there is an operational tunnel interface configured with the tunnel source and the destination address that matches the incoming packets' outer source and destination addresses. If there is not a source and destination packet match or if the interface is in shutdown state, the packet is dropped.

  Decapsulate Tunnel (Source only) - Cisco Nexus 3000 Series switches decapsulate all IP-in-IP packets destined to it when the command **feature tunnel** is configured and there is an operational tunnel interface configured with the tunnel source address that matches the incoming packets' outer destination addresses. If there is not a source packet match or if the interface is in shutdown state, the packet is dropped.
- Starting with Release 6.0(2)U6(1), Cisco Nexus 3000 Series switches support IPv6 in IPv4 with GRE header only. The new control protocols that are supported on the tunnel are:
  - BGP with v6
  - OSPFv3
  - EIGRP over v6
- The Cisco Nexus 3000 Series switches ASIC supports the GRE encapsulation and decapsulation in the hardware.
- On the encapsulation side, the Cisco Nexus 3000 Series switches performs a single lookup in the hardware.
- Since Cisco Nexus 3000 Series switches perform a single lookup in the hardware, the software has to keep the hardware information up-to-date with any changes related to the second lookup, for example, the tunnel destination adjacency.
- On the decapsulation side, the Cisco Nexus 3000 Series switches have a separate table to perform the outer IP header lookup and it does not need an ACL for the same.

# Default Settings for IP Tunneling

The following table lists the default settings for IP tunnel parameters.

***Table 6: Default IP Tunnel Parameters***

| Parameters | Default |
|---|---|
| Tunnel feature | Disabled |

# Configuring IP Tunnels

## Enabling Tunneling

### Before You Begin

You must enable the tunneling feature before you can configure any IP tunnels.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **feature tunnel** | Enables the tunnel feature on the switch. |
| **Step 3** | switch(config)# **exit** | Returns to configuration mode. |
| **Step 4** | switch(config)# **show feature** | Displays the tunnel feature on the switch. |
| **Step 5** | switch# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to enable the tunnel feature:

```
switch# configure terminal
switch(config)# feature tunnel
switch(config)# exit
switch(config)# copy running-config startup-config
```

## Creating a Tunnel Interface

You can create a tunnel interface and then configure this logical interface for your IP tunnel. GRE mode is the default tunnel mode.

### Before You Begin

Both the tunnel source and the tunnel destination must exist within the same virtual routing and forwarding (VRF) instance.

Ensure that you have enabled the tunneling feature.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# [**no**] **interface tunnel** *number* | Creates a new tunnel interface. |
| **Step 3** | switch(config)# **tunnel mode** {**gre ip** \| **ipip** {**ip** \| **decapsulate-any**}} | Sets this tunnel mode to GRE, ipip, or ipip decapsulate-only.<br>The **gre** and **ip** keywords specify that GRE encapsulation over IP will be used.<br>The **ipip** keyword specifies that IP-in-IP encapsulation will be used. The optional **decapsulate-any** keyword terminates IP-in-IP tunnels at one tunnel interface. This keyword creates a tunnel that will not carry any outbound traffic. However, remote tunnel endpoints can use a tunnel configured as their destination. |
| **Step 4** | switch(config)# **tunnel source** {*ip address* \| *interface-name*} | Configures the source address for this IP tunnel. |
| **Step 5** | switch(config)# **tunnel destination** {*ip address* \| *host-name*} | Configures the destination address for this IP tunnel. |
| **Step 6** | switch(config)# **tunnel use-vrf** *vrf-name* | (Optional)<br>Uses the configured VRF instance to look up the tunnel IP destination address. |
| **Step 7** | switch(config)# **show interface tunnel** *number* | (Optional)<br>Displays the tunnel interface statistics. |
| **Step 8** | switch# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to create a tunnel interface:

```
switch# configure terminal
switch(config)# interface tunnel 1
switch(config)# tunnel source ethernet 1/2
switch(config)# tunnel destination 192.0.2.1
switch(config)# copy running-config startup-config
```

# Configuring a Tunnel Interface Based on Policy Based Routing

You can create a tunnel interface and then configure this logical interface for your IP tunnel based on the PBR policy.

### Before You Begin

Ensure that you have enabled the tunneling feature.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# [**no**] **interface tunnel** *number* | Creates a new tunnel interface. |
| **Step 3** | switch(config)# **ip address** *ip address* | Configures an IP address for this interface. |
| **Step 4** | switch(config)# **route-map** *map-name* | Assigns a route map for IPv4 policy-based routing to the interface |
| **Step 5** | switch(config-route-map)# **match ip address access-list-name** *name* | Matches an IPv4 address against one or more IP access control lists (ACLs). This command is used for policy-based routing and is ignored by route filtering or redistribution. |
| **Step 6** | switch(config-route-map)# **set ip next-hop** *address* | Sets the IPv4 next-hop address for policy-based routing. Tunnel IP addresses can be specified as next-hop addresses to select tunnel interfaces. This command uses the first valid next-hop address if multiple addresses are configured. Use the **load-share** option to select ECMP across next-hop entries. |

This example shows how to configure a tunnel interface that is based on PBR:

```
switch# configure terminal
switch(config)# interface tunnel 1
switch(config)# ip address 1.1.1.1/24
switch(config)# route-map pbr1
switch(config-route-map)# match ip address access-list-name pbr1
switch(config-route-map)# set ip next-hop 1.1.1.1
```

# Configuring a GRE Tunnel

You can set a tunnel interface to GRE tunnel mode, ipip mode, or ipip decapsulate-only mode. GRE mode is the default tunnel mode. Starting with Release 6.0(2)U6(1), Cisco Nexus 3000 Series switches support IPv6 in IPv4 with GRE header only.

### Before You Begin

Ensure that you have enabled the tunneling feature.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface tunnel** *number* | Enters a tunnel interface configuration mode. |
| **Step 3** | switch(config-if)# **tunnel mode** {**gre ip** \| **ipip** {**ip** \| **decapsulate-any**}} | Sets this tunnel mode to GRE, ipip, or ipip decapsulate-only.<br>The **gre** and **ip** keywords specify that GRE encapsulation over IP will be used.<br>The ipip keyword specifies that IP-in-IP encapsulation will be used. The optional **decapsulate-any** keyword terminates IP-in-IP tunnels at one tunnel interface. This keyword creates a tunnel that will not carry any outbound traffic. However, remote tunnel endpoints can use a tunnel configured as their destination. |
| **Step 4** | switch(config-if)# **tunnel use-vrf** *vrf-name* | Configures tunnel VRF name. |
| **Step 5** | switch(config-if)# **ipv6 address** *IPv6 address* | Configures the IPv6 address.<br>**Note** The tunnel source and the destination addresses are still the same (IPv4 address.) |
| **Step 6** | switch(config-if)# **show interface tunnel** *number* | (Optional)<br>Displays the tunnel interface statistics. |
| **Step 7** | switch(config-if)# **mtu** *value* | Sets the maximum transmission unit (MTU) of IP packets sent on an interface. |
| **Step 8** | switch(config-if)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to create the tunnel interface to GRE:

```
switch# configure terminal
switch(config)# interface tunnel 1
switch(config-if)# tunnel mode gre ip
switch(config-if)# tunnel use-vrf red
switch(config-if)# ipv6 address 2:2::2/64
switch(config-if)# copy running-config startup-config
```

This example shows how to view the tunnel interface to GRE:

```
switch(config)# show int tunnel 2
Tunnel2 is up
    Internet address(es):
    2.2.2.2/24
    2:2::2/64
    MTU 1476 bytes, BW 9 Kbit
    Transport protocol is in VRF "default"
    Tunnel protocol/transport GRE/IP
```

```
    Tunnel source 2.2.3.2, destination 2.2.3.1
    Last clearing of "show interface" counters never
    Tx
    0 packets output, 0 bytes
```

This example shows how to create an ipip tunnel:

```
switch# configure terminal
switch(config)# interface tunnel 1
switch(config-if)# tunnel mode ipip
switch(config-if)# mtu 1400
switch(config-if)# copy running-config startup-config
switch(config-if)# no shut
```

# Assigning VRF Membership to a Tunnel Interface

You can add a tunnel interface to a VRF.

### Before You Begin

Ensure that you have enabled the tunneling feature.

Assign the IP address for a tunnel interface after you have configured the interface for a VRF.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface tunnel** *number* | Enters interface configuration mode. |
| **Step 3** | switch(config)# **vrf member** *vrf-name* | Adds this interface to a VRF. |
| **Step 4** | switch(config)# **ip address** *ip-prefix/length* | Configures an IP address for this interface. You must do this step after you assign this interface to a VRF. |
| **Step 5** | switch(config)# **show vrf** [*vrf-name*] **interface** *interface-type number* | (Optional)<br>Displays VRF information. |
| **Step 6** | switch(config-if)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to add a tunnel interface to the VRF:

```
switch# configure terminal
switch(config)# interface tunnel 0
switch(config-if)# vrf member RemoteOfficeVRF
switch(config-if)# ip address 209.0.2.1/16
switch(config-if)# copy running-config startup-config
```

# Verifying the IP Tunnel Configuration

Use the following commands to verify the configuration:

| Command | Purpose |
|---|---|
| **show interface tunnel** *number* | Displays the configuration for the tunnel interface (MTU, protocol, transport, and VRF). Displays input and output packets, bytes, and packet rates. |
| **show interface tunnel** *number* **brief** | Displays the operational status, IP address, encapsulation type, and MTU of the tunnel interface. |
| **show interface tunnel** *number* **description** | Displays the configured description of the tunnel interface. |
| **show interface tunnel** *number* **status** | Displays the operational status of the tunnel interface. |
| **show interface tunnel** *number* **status err-disabled** | Displays the error disabled status of the tunnel interface. |

# Configuration Examples for IP Tunneling

This example shows a simple GRE tunnel. Ethernet 1/2 is the tunnel source for router A and the tunnel destination for router B. Ethernet interface 1/3 is the tunnel source for router B and the tunnel destination for router A.

```
router A:
feature tunnel
interface tunnel 0
  ip address 209.165.20.2/8
  tunnel source ethernet 1/2
  tunnel destination 192.0.2.2
  tunnel mode gre ip
interface ethernet1/2
  ip address 192.0.2.55/8

router B:
feature tunnel
interface tunnel 0
  ip address 209.165.20.1/8
  tunnel source ethernet 1/3
  tunnel destination 192.0.2.55
  tunnel mode gre ip
interface ethernet 1/3
ip address 192.0.2.2/8
```

# Related Documents for IP Tunnels

| Related Topics | Document Title |
|---|---|
| IP tunnel commands | *Cisco Nexus 3000 Series Interfaces Command Reference* |

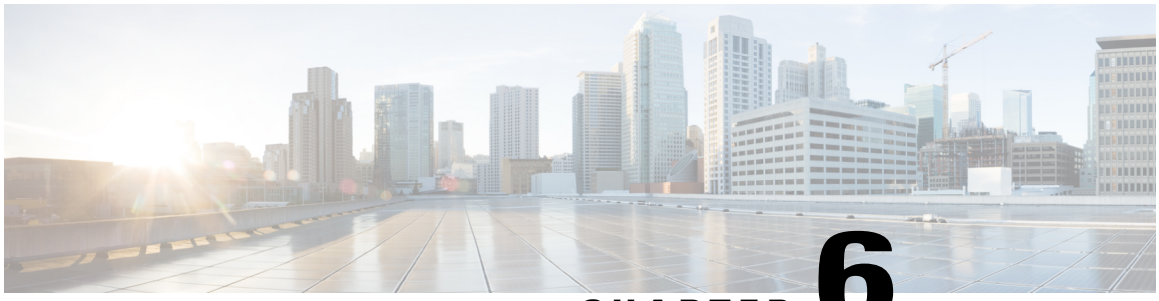
# Standards for IP Tunnels

No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.

# Feature History for Configuring IP Tunnels

***Table 7: Feature History for Configuring IP Tunnels***

| Feature Name | Release | Feature Information |
|---|---|---|
| Multi-point and Point-to-Point IP-in-IP encapsulation and decapsulation | 6.0(2)U2(1) | Support for these tunnel modes was added. |
| IP tunnels | 5.0(3)U4(1) | This feature was introduced. |



CHAPTER 6

# Configuring VXLANs

This chapter contains the following sections:



# Overview

## VXLAN Overview

The Cisco Nexus 3100 Series switches are designed for a hardware-based Virtual Extensible LAN (VXLAN) function. These switches can extend Layer 2 connectivity across the Layer 3 boundary and integrate between VXLAN and non-VXLAN infrastructures. Virtualized and multitenant data center designs can be shared over a common physical infrastructure.

VXLANs enable you to extend Layer 2 networks across the Layer 3 infrastructure by using MAC-in-UDP encapsulation and tunneling. In addition, you can use a VXLAN to build a multitenant data center by decoupling tenant Layer 2 segments from the shared transport network.

When deployed as a VXLAN gateway, the Cisco Nexus 3100 Series switches can connect VXLAN and classic VLAN segments to create a common forwarding domain so that tenant devices can reside in both environments.

A VXLAN has the following benefits:

- Flexible placement of multitenant segments throughout the data center.

  It extends Layer 2 segments over the underlying shared network infrastructure so that tenant workloads can be placed across physical pods in the data center.

- Higher scalability to address more Layer 2 segments.

A VXLAN uses a 24-bit segment ID called the VXLAN network identifier (VNID). The VNID allows a maximum of 16 million VXLAN segments to coexist in the same administrative domain. (In comparison, traditional VLANs use a 12-bit segment ID that can support a maximum of 4096 VLANs.)

- Utilization of available network paths in the underlying infrastructure.

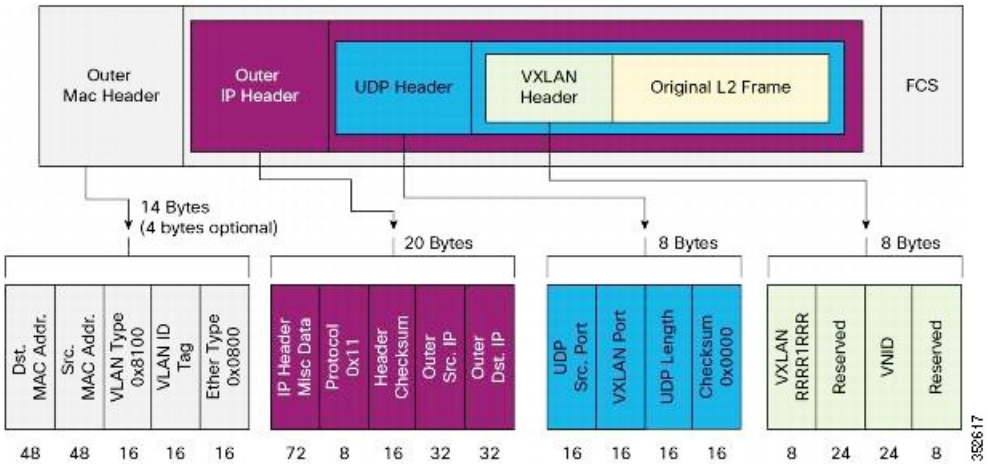
  VXLAN packets are transferred through the underlying network based on its Layer 3 header. It uses equal-cost multipath (ECMP) routing and link aggregation protocols to use all available paths.

# VXLAN Encapsulation and Packet Format

A VXLAN is a Layer 2 overlay scheme over a Layer 3 network. It uses MAC-in-UDP encapsulation to extend Layer 2 segments across the data center network. The transport protocol over the physical data center network is IP plus UDP.

A VXLAN defines a MAC-in-UDP encapsulation scheme where the original Layer 2 frame has a VXLAN header added and is then placed in a UDP-IP packet. With this MAC-in-UDP encapsulation, VXLAN tunnels Layer 2 network over the Layer 3 network. The VXLAN packet format is shown in the following figure.

***Figure 6: VXLAN Packet Format***




A VXLAN uses an 8-byte VXLAN header that consists of a 24-bit VNID and a few reserved bits. The VXLAN header and the original Ethernet frame are in the UDP payload. The 24-bit VNID identifies the Layer 2 segments and maintains Layer 2 isolation between the segments. A VXLAN can support 16 million LAN segments.

# VXLAN Tunnel Endpoints

A VXLAN uses VXLAN tunnel endpoint (VTEP) devices to map tenants' end devices to VXLAN segments and to perform VXLAN encapsulation and deencapsulation. Each VTEP device has two types of interfaces:

- Switch port interfaces on the local LAN segment to support local endpoint communication through bridging
- IP interfaces to the transport network where the VXLAN encapsulated frames will be sent

A VTEP device is identified in the IP transport network by using a unique IP address, which is a loopback interface IP address. The VTEP device uses this IP address to encapsulate Ethernet frames and transmits the encapsulated packets to the transport network through the IP interface. A VTEP device learns the remote VTEP IP addresses and the remote MAC address-to-VTEP IP mapping for the VXLAN traffic that it receives.

The VXLAN segments are independent of the underlying network topology; conversely, the underlying IP network between VTEPs is independent of the VXLAN overlay. The IP network routes the encapsulated packets based on the outer IP address header, which has the initiating VTEP as the source IP address and the terminating VTEP or multicast group IP address as the destination IP address.

# VXLAN Packet Forwarding Flow

A VXLAN uses stateless tunnels between VTEPs to transmit traffic of the overlay Layer 2 network through the Layer 3 transport network.

# VXLAN Implementation on Cisco Nexus 3100 Series Switches

The Cisco Nexus 3100 Series switches support the hardware-based VXLAN function that extends Layer 2 connectivity across the Layer 3 transport network and provides a high-performance gateway between VXLAN and non-VXLAN infrastructures.

# Layer 2 Mechanisms for Broadcast, Unknown Unicast, and Multicast Traffic

A VXLAN on the Cisco Nexus 3100 Series switches uses flooding and dynamic MAC address learning to do the following:

- Transport broadcast, unknown unicast, and multicast traffic
- Discover remote VTEPs
- Learn remote host MAC addresses and MAC-to-VTEP mappings for each VXLAN segment

A VXLAN can forward these traffic types as follows:

- Using multicast in the core—IP multicast reduces the flooding of the set of hosts that are participating in the VXLAN segment. Each VXLAN segment, or VNID, is mapped to an IP multicast group in the transport IP network. The Layer 2 gateway uses Protocol Independent Multicast (PIM) to send and receive traffic from the rendezvous point (RP) for the IP multicast group. The multicast distribution tree for this group is built through the transport network based on the locations of participating VTEPs.
- Using ingress replication—Each VXLAN segment or VXLAN network identifier (VNI) is mapped to a remote unicast peer. The Layer 2 frame is VXLAN encapsulated with the destination IP address as the remote unicast peer IP address and is sent out to the IP transport network where it gets unicast routed or forwarded to the remote destination.

# Layer 2 Mechanisms for Unicast-Learned Traffic

The Cisco Nexus 3100 Series switches perform MAC address lookup-based forwarding for VXLAN unicast-learned traffic.

When Layer 2 traffic is received on the access side, a MAC address lookup is performed for the destination MAC address in the frame. If the lookup is successful, VXLAN forwarding is done based on the information retrieved as a result of the lookup. The lookup result provides the IP address of the remote VTEP from which this MAC address is learned. This Layer 2 frame is then UDP/IP encapsulated with the destination IP address as the remote VTEP IP address and is forwarded out of the appropriate network interface. In the Layer 3 cloud, this IP packet is forwarded to the remote VTEP through the route to that IP address in the network.

For unicast-learned traffic, you must ensure the following:

- The route to the remote peer is known through a routing protocol or through static routes in the network.
- Adjacency is resolved.

# VXLAN Layer 2 Gateway as a Transit Multicast Router

A VXLAN Layer 2 gateway must terminate VXLAN-multicast traffic that is headed to any of the groups to which VNIs are mapped. In a network, a VXLAN Layer 2 gateway can be a multicast transit router for the downstream multicast receivers that are interested in the group's traffic. A VXLAN Layer 2 gateway must do some additional processing to ensure that VXLAN multicast traffic that is received is both terminated and multicast routed. This traffic processing is done in two passes:

1. The VXLAN multicast traffic is multicast routed to all network receivers interested in that group's traffic.
2. The VXLAN multicast traffic is terminated, decapsulated, and forwarded to all VXLAN access side ports.

# ECMP and LACP Load Sharing with VXLANs

Encapsulated VXLAN packets are forwarded between VTEPs based on the native forwarding decisions of the transport network. Most data center transport networks are designed and deployed with multiple redundant paths that take advantage of various multipath load-sharing technologies to distribute traffic loads on all available paths.

A typical VXLAN transport network is an IP-routing network that uses the standard IP equal cost multipath (ECMP) to balance the traffic load among multiple best paths. To avoid out-of-sequence packet forwarding, flow-based ECMP is commonly deployed. An ECMP flow is defined by the source and destination IP addresses and optionally, the source and destination TCP or UDP ports in the IP packet header.

All the VXLAN packet flows between a pair of VTEPs have the same outer source and destination IP addresses, and all VTEP devices must use one identical destination UDP port that can be either the Internet Allocated Numbers Authority (IANA)-allocated UDP port 4789 or a customer-configured port. The only variable element in the ECMP flow definition that can differentiate VXLAN flows from the transport network standpoint is the source UDP port. A similar situation for Link Aggregation Control Protocol (LACP) hashing occurs if the resolved egress interface that is based on the routing and ECMP decision is an LACP port channel. LACP uses the VXLAN outer-packet header for link load-share hashing, which results in the source UDP port being the only element that can uniquely identify a VXLAN flow.

In the Cisco Nexus 3100 Series switches implementation of VXLANs, a hash of the inner frame's header is used as the VXLAN source UDP port. As a result, a VXLAN flow can be unique. The IP address and UDP port combination is in its outer header while the packet traverses the underlay transport network.

# Guidelines and Limitations for VXLANs

VXLAN has the following guidelines and limitations:

- The Cisco Nexus 3100 Series switches do not support VTEP under vPC.
- The Cisco Nexus 3100 series switches do not support anycast RP with VxLAN.
- IGMP snooping is not supported on VXLAN VLANs.
- VXLAN routing is not supported. The default Layer 3 gateway for VXLAN VLANs must be provisioned on a different device.
- Ensure that the network can accommodate an additional 50 bytes for the VXLAN header.
- Only one Network Virtualization Edge (NVE) interface is supported on a switch.
- Layer 3 VXLAN uplinks are not supported in a nondefault virtual and routing forwarding (VRF) instance.
- Only one VXLAN IP adjacency is possible per physical interface.
- Switched virtual interfaces (SVIs) are not supported on VXLAN VLANs.
- Switched Port Analyzer (SPAN) Tx for VXLAN-encapsulated traffic is not supported for the Layer 3 uplink interface.
- Access control lists (ACLs) and quality of service (QoS) for VXLAN traffic to access direction are not supported.
- SNMP is not supported on the NVE interface.
- Native VLANs for VXLAN are not supported.
- For ingress replication configurations, multiple VNIs can now have the same remote peer IP configured.
- Use the **ip pim spt-threshold infinity group-list** command to ensure that Shortest Path Tree (SPT) is not selected for the VXLAN multicast group.
- The VXLAN source UDP port is determined based on the VNID and source and destination IP addresses.
- The UDP port configuration must be done before the NVE interface is enabled. If the UDP configuration must be changed while the NVE interface is enabled, you must shut down the NVE interface, make the UDP configuration change, and then reenable the NVE interface.
- When a VN-Segment is mapped to a native VLAN, if traffic is sent on any normal VLAN on that port instead of getting switched in the VLAN, it gets forwarded in the VXLAN tunnel for the native VLAN.

# Considerations for VXLAN Deployment

The following are some of the considerations while deploying VXLANs:

- A loopback interface IP is used to uniquely identify a VTEP device in the transport network.
- To establish IP multicast routing in the core, an IP multicast configuration, PIM configuration, and Rendezvous Point (RP) configuration are required.
- You can configure VTEP-to-VTEP unicast reachability through any IGP protocol.

- You can configure a VXLAN UDP destination port as required. The default port is 4789.
- The default gateway for VXLAN VLANs should be provisioned on a different upstream router.
- VXLAN multicast traffic should always use the RPT shared tree.
- An RP for the multicast group on the VTEP is a supported configuration. However, you must configure the RP for the multicast group at the spine layer/upstream device. Because all multicast traffic traverses the RP, it is more efficient to have this traffic directed to a spine layer/upstream device.

# vPC Guidelines and Limitations for VXLAN Deployment

- You must bind NVE to a loopback address that is separate form other loopback addresses required by Layer 3 protocols. Use a dedicated loopback address for VXLAN.
- vPC peers must have identical configurations for the following:
  - Consistent VLAN to VN-Segment mapping.
  - Consistent NVE binding:
    - Using the same source interface IP address.
    - Using consistent VNI to group mapping.
- For multicast, the vPC node that receives the (S,G) join from the RP becomes the designated forwarder (DF). On the DF node, both encapsulation and decapsulation routes are installed for multicast. The other vPC node does not initiate or terminate multicast traffic.
- Multicast traffic on a vPC that is hashed toward the non-DF switch traverses the multichassis EtherChannel trunk (MCT) and is encapsulated on the DF node.
- When MCT is shut, the loopback interface on the secondary vPC is brought down and the status is **Admin Shut**. The route to the loopback is withdrawn on the upstream and the upstream can divert all traffic to the primary vPC.

**Note** Orphans that are connected to the secondary vPC experience a loss of traffic when the MCT is shut down. This situation is similar to Layer 2 orphans in a secondary vPC of a traditional vPC setup.

- In a VXLAN vPC, consistency checks are performed to ensure that NVE configurations and VN-Segment configurations are identical across vPC peers.
- The router ID for unicast routing protocols must be different from the loopback IP address used for VTEP.
- When MCT is no-shut, the NVE loopback interface is brought up again and the route is advertised upstream to attract traffic.
- Configure an SVI between vPC peers and advertise routes between the vPC peers by using a routing protocol with higher routing metric. This action ensures that the IP connectivity of the vPC node does not go down if one vPC node fails.

# Configuring VXLAN Traffic Forwarding

There are two options for forwarding broadcast, unknown unicast and multicast traffic on a VXLAN Layer 2 gateway. Layer 2 Mechanisms for Broadcast, Unknown Unicast, and Multicast Traffic, on page 89 provides more information about these two options.

Before you enable and configure VXLANs, ensure that the following configurations are complete:

- For IP multicast in the core, ensure that the IP multicast configuration, the PIM configuration, and the RP configuration are complete, and that a routing protocol exists.
- For ingress replication, ensure that a routing protocol exists for reaching unicast addresses.

**Note** On a Cisco Nexus 3100 Series switch that functions as a VXLAN Layer 2 gateway, note that traffic that is received on the access side cannot trigger an ARP on the network side. ARP for network side interfaces should be resolved either by using a routing protocol such as BGP, or by using static ARP. This requirement is applicable for ingress replication cases alone, not for multicast replication cases.

# Enabling and Configuring the PIM Feature

Before you can access the PIM commands, you must enable the PIM feature.

This is a prerequisite only for multicast replication.

### Before You Begin

Ensure that you have installed the LAN Base Services license.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **feature pim** | Enables PIM. By default, PIM is disabled. |
| **Step 3** | switch(config)# **ip pim spt-threshold infinity group-list** *route-map-name* | Creates the IPv4 Protocol Independent Multicast (PIM) (*, G) state only. Allows selection of the RPT only and not the SPT. |
| **Step 4** | switch(config)# **show running-config pim** | (Optional)<br>Shows the running-configuration information for PIM, including the **feature** command. |
| **Step 5** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to enable the PIM feature:

```
switch# configure terminal
switch(config)# feature pim
switch(config)# ip pim spt-threshold infinity group-list rp_name
switch(config)# show running-config pim

!Command: show running-config pim
!Time: Wed Mar 26 08:04:23 2014

version 6.0(2)U3(1)
feature pim

ip pim spt-threshold infinity group-list rp_name
```

# Configuring a Rendezvous Point

You can configure a rendezvous point (RP) by configuring the RP address on every router that will participate in the PIM domain.

This is a prerequisite only for multicast replication.

### Before You Begin

Ensure that you have installed the LAN Base Services license and enabled PIM.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **ip pim rp-address** *rp-address* [**group-list** *ip-prefix* \| **route-map** *policy-name*] | Configures a PIM RP address for a multicast group range. You can specify a route-map policy name that lists the group prefixes to use with the **match ip multicast** command. The default mode is ASM. The default group range is 224.0.0.0 through 239.255.255.255. |
| **Step 3** | switch(config)# **show ip pim group-range** [*ip-prefix*] [**vrf** {*vrf-name* \| **all** \| **default** \| **management**}] | (Optional)<br>Displays PIM modes and group ranges. |
| **Step 4** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to configure an RP:

```
switch# configure terminal
switch(config)# ip pim rp-address 111.1.1.1 group-list 224.0.0.0/4
```

# Enabling a VXLAN

Enabling VXLANs involves the following:

- Enabling the VXLAN feature
- Enabling VLAN to VN-Segment mapping

### Before You Begin

Ensure that you have installed the VXLAN Enterprise license.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **[no] feature nv overlay** | Enables the VXLAN feature. |
| **Step 3** | switch (config)# **[no] feature vn-segment-vlan-based** | Configures the global mode for all VXLAN bridge domains.<br>Enables VLAN to VN-Segment mapping. VLAN to VN-Segment mapping is always one-to-one. |
| **Step 4** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to enable a VXLAN and configure VLAN to VN-Segment mapping:

```
switch# configure terminal
switch(config)# feature nv overlay
switch(config)# feature vn-segment-vlan-based
switch(config)# copy running-config startup-config
```

# Mapping a VLAN to a VXLAN VNI

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **vlan** *vlan-id* | Specifies a VLAN. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | switch(config-vlan)# **vn-segment** *vnid* | Specifies the VXLAN virtual network identifier (VNID). |

This example shows how to map a VLAN to a VXLAN VNI:

```
switch# configure terminal
switch(config)# vlan 3100
switch(config-vlan)# vn-segment 5000
```

# Configuring a Routing Protocol for NVE Unicast Addresses

Configuring a routing protocol for unicast addresses involves the following:

- Configuring a dedicated loopback interface for NVE reachability.
- Configuring the routing protocol network type.
- Specifying the routing protocol instance and area for an interface.
- Enabling PIM sparse mode in case of multicast replication.

**Note** Open shortest path first (OSPF) is used as the routing protocol in the examples.

This is a prerequisite for both multicast and ingress replication.

Guidelines for configuring a routing protocol for unicast addresses are as follows:

- For ingress replication, you can use a routing protocol that can resolve adjacency, such as BGP.
- When using unicast routing protocols in a vPC topology, explicitly configure a unique router ID for the vPC peers to avoid the VTEP loopback IP address (which is the same on the vPC peers) being used as the router ID.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface loopback** *instance* | Creates a dedicated loopback interface for the NVE interface. The instance range is from 0 to 1023. |
| **Step 3** | switch(config-if)# **ip address** *ip-address/length* | Configures an IP address for this interface. |
| **Step 4** | switch(config-if)# **ip ospf network** {**broadcast** \| **point-to-point**} | Configures the OSPF network type to a type other than the default for an interface. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | switch(config-if)# **ip router ospf** *instance-tag* **area** *area-id* | Specifies the OSPF instance and area for an interface. |
| **Step 6** | switch(config-if)# **ip pim sparse-mode** | Enables PIM sparse mode on this interface. The default is disabled.<br>Enable the PIM sparse mode in case of multicast replication. |

This example shows how to configure a routing protocol for NVE unicast addresses:

```
switch# configure terminal
switch(config)# interface loopback 10
switch(config-if)# ip address 222.2.2.1/32
switch(config-if)# ip ospf network point-to-point
switch(config-if)# ip router ospf 1 area 0.0.0.0
switch(config-if)# ip pim sparse-mode
```

# Creating a VXLAN Destination UDP Port

The UDP port configuration should be done before the NVE interface is enabled.

**Note** If the configuration must be changed while the NVE interface is enabled, ensure that you shut down the NVE interface, make the UDP configuration change, and then reenable the NVE interface.

Ensure that the UDP port configuration is done network-wide before the NVE interface is enabled on the network.

The VXLAN UDP source port is determined based on the VNID and source and destination IP addresses.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **vxlan udp port** *number* | Specifies the destination UDP port number for VXLAN encapsulated packets. The default destination UDP port number is 4789. |

This example shows how to create a VXLAN destination UDP port:

```
switch# configure terminal
switch(config)# vxlan udp port 4789
```

# Creating and Configuring an NVE Interface

An NVE interface is the overlay interface that initiates and terminates VXLAN tunnels. You can create and configure an NVE (overlay) interface.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface nve** *instance* | Creates a VXLAN overlay interface that initiates and terminates VXLAN tunnels.<br>**Note** Only one NVE interface is allowed on the switch. |
| **Step 3** | switch(config-if-nve)# **source-interface loopback** *instance* | Specifies a source interface.<br>The source interface must be a loopback interface that is configured on the switch with a valid /32 IP address. This /32 IP address must be known by the transit routers in the transport network and the remote VTEPs. |

This example shows how to create and configure an NVE interface:

```
switch# configure terminal
switch(config)# interface nve 1
switch(config-if-nve)# source-interface loopback 10
```

# Configuring Replication for a VNI

Replication for VXLAN network identifier (VNI) can be configured in one of two ways:

- Multicast replication
- Ingress replication

## Configuring Multicast Replication

### Before You Begin

- Ensure that the NVE interface is created and configured.
- Ensure that the source interface is specified.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch(config-if-nve)# **member vni** {*vnid* **mcast-group** *multicast-group-addr* \| *vnid- range* **mcast-group** *start-addr* [*end-addr*]} | Maps VXLAN VNIs to the NVE interface and assigns a multicast group to the VNIs. |

This example shows how to map a VNI to an NVE interface and assign it to a multicast group:

```
switch(config-if-nve)# member vni 5000 mcast-group 225.1.1.1
```

## Configuring Ingress Replication

### Before You Begin

- Ensure that the NVE interface is created and configured.
- Ensure that the source interface is specified.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch(config-if-nve)# **member vni** *vnid* | Maps VXLAN VNIs to the NVE interface. |
| **Step 2** | switch(config-if-nve-vni)# **ingress-replication** *ip-address* | Enables ingress replication for the VNI to the specified unicast address.<br>**Note** • A VNI can be associated only with a single IP address.<br>• An IP address can be associated only with a single VNI. |

This example shows how to map a VNI to an NVE interface and create a unicast tunnel:

```
switch(config-if-nve)# member vni 5001
switch(config-if-nve-vni)# ingress-replication 111.1.1.1
```

# Verifying the VXLAN Configuration

Use one of the following commands to verify the VXLAN configuration:

| Command | Purpose |
|---|---|
| **show nve interface nve** *id* | Displays the configuration of an NVE interface. |

| Command | Purpose |
|---|---|
| **show nve vni** | Displays the VNI that is mapped to an NVE interface. |
| **show nve peers** | Displays peers of the NVE interface. |
| **show interface nve** *id* **counters** | Displays all the counters for an NVE interface. |
| **show nve vxlan-params** | Displays the VXLAN UDP port configured. |

This example shows how to display the configuration of an NVE interface:

```
switch# show nve interface nve 1
Interface: nve1, State: up, encapsulation: VXLAN
Source-interface: loopback10 (primary: 111.1.1.1, secondary: 0.0.0.0)
```

This example shows how to display the VNI that is mapped to an NVE interface for multicast replication:

```
switch# show nve vni
Interface          VNI        Multicast-group   VNI State
----------------   --------   ---------------   ---------
nve1               5000       225.1.1.1          Up
```

This example shows how to display the VNI that is mapped to an NVE interface for ingress replication:

```
switch# show nve vni
Interface          VNI        Multicast-group   VNI State
----------------   --------   ---------------   ---------
nve1               5000       0.0.0.0           Up
```

This example shows how to display the peers of an NVE interface:

```
switch# show nve peers
Interface          Peer-IP           Peer-State
----------------   ---------------   -------------
nve1               111.1.1.1          Up
```

This example shows how to display the counters of an NVE interface:

```
switch# show interface nv 1 counter

--------------------------------------------------------------------------------
Port                                  InOctets                    InUcastPkts
--------------------------------------------------------------------------------
nve1                                         0                              0

--------------------------------------------------------------------------------
Port                                InMcastPkts                    InBcastPkts
--------------------------------------------------------------------------------
nve1                                         0                              0

--------------------------------------------------------------------------------
Port                                 OutOctets                   OutUcastPkts
--------------------------------------------------------------------------------
nve1                                         0                              0

--------------------------------------------------------------------------------
Port                               OutMcastPkts                   OutBcastPkts
--------------------------------------------------------------------------------
nve1                                         0                              0
```

This example shows how to display the VXLAN UDP port configured:

```
switch# show nve vxlan-params
VxLAN Dest. UDP Port: 4789
```

# Displaying MAC Addresses

Enter one of these commands to display VXLAN and VLAN MAC addresses:

| Command | Purpose |
|---|---|
| **show mac address-table** | Displays both VLAN and VXLAN MAC addresses. |
| **show mac address-table vlan** *vlan-id* | Displays all the VxLAN MAC addresses that are learned on the specified VLAN. For VN-Segment mapped VLANs, it displays both local and remote MAC addresses. |
| **show mac address-table local** | Displays only locally learned MAC addresses on all VLANs that are mapped to VN-Segments. |
| **show mac address-table local vlan** *vlan-id* | Displays only locally learned MAC addresses on the specified VLAN, which is mapped to a VN-Segment. |
| **show mac address-table interface nve** *nve-id* | Displays all remote MAC addresses learned on NVE. |
| **show mac address-table interface nve** *nve-id* **vni** *vni-id* | Displays all remote MAC addresses learned on the VNI. |
| **show mac address-table interface ethernet** *slot/port* **vlan** *vlan-id* | Displays all MAC addresses learned on the VLAN on this interface. |
| **show mac address-table interface nve** *nve-id* **peer** *ip-address*<br>**show mac address-table interface nve** *nve-id* **peer vrf** *vrf-name ip-address* | Displays all MAC addresses learned on NVE from the specified peer. |
| **show mac address-table interface nve** *nve-id* **peer** *ip-address* **vni** *vni-id*<br>**show mac address-table interface nve** *nve-id* **peer vrf** *vrf-name ip-address* **vni** *vni-id* | Displays all MAC addresses learned on NVE from the specified peer on the specified VNI. |
| **show mac address-table count local** | Displays the number of locally learned MAC address table entries. |
| **show mac address-table count local vlan** *vlan-id* | Displays the number of locally learned MAC address table entries on the specified VLAN, which is mapped to a VN-segment. |

| Command | Purpose |
|---|---|
| **show mac address-table count interface nve** *nve-id* | Displays the number of remote MAC address table entries learned on NVE. |
| **show mac address-table count interface nve** *nve-id* **vni** *vni-id* | Displays the number of remote MAC address table entries learned on the VNI. |
| **show mac address-table count interface nve** *nve-id* **peer** *ip-address*<br>**show mac address-table count interface nve** *nve-id* **peer** *ip-address* **vrf** *vrf-name* | Displays the number of MAC address table entries learned on NVE from the specified peer. |
| **show mac address-table count interface nve** *nve-id* **peer** *ip-address* **vni** *vni-id*<br>**show mac address-table count interface nve** *nve-id* **peer** *ip-address* **vrf** *vrf-name* **vni** *vni-id* | Displays the number of MAC address table entries learned on NVE from the specified peer on the specified VNI. |

This example shows how to display both VLAN and VXLAN MAC addresses:

```
switch# show mac address-table
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since first seen,+ - primary entry using vPC Peer-Link
   VLAN     MAC Address      Type      age     Secure NTFY   Ports/SWID.SSID.LID
---------+-----------------+--------+---------+------+----+------------------
* 109      0000.0410.0902    dynamic   470        F    F  Po2233
* 109      0000.0410.0912    dynamic   470        F    F  Po2233
* 109      0000.0410.0912    dynamic   470        F    F  nve1(1.1.1.200)
* 108      0000.0410.0802    dynamic   470        F    F  Po2233
* 108      0000.0410.0812    dynamic   470        F    F  Po2233
* 107      0000.0410.0702    dynamic   470        F    F  Po2233
* 107      0000.0410.0712    dynamic   470        F    F  Po2233
* 107      0000.0410.0712    dynamic   470        F    F  nve1(1.1.1.200)
* 106      0000.0410.0602    dynamic   470        F    F  Po2233
* 106      0000.0410.0612    dynamic   470        F    F  Po2233
* 105      0000.0410.0502    dynamic   470        F    F  Po2233
* 105      0000.0410.0512    dynamic   470        F    F  Po2233
* 105      0000.0410.0512    dynamic   470        F    F  nve1(1.1.1.200)
* 104      0000.0410.0402    dynamic   470        F    F  Po2233
* 104      0000.0410.0412    dynamic   470        F    F  Po2233
```

This example shows how to display all the VXLAN MAC addresses learned on the specified VLAN:

```
switch# show mac address-table vlan 107
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since first seen,+ - primary entry using vPC Peer-Link
   VLAN     MAC Address      Type      age     Secure NTFY   Ports/SWID.SSID.LID
---------+-----------------+--------+---------+------+----+------------------
* 107      0000.0410.0702    dynamic   470        F    F  Po2233
* 107      0000.0410.0712    dynamic   470        F    F  Po2233
* 107      0000.0410.0712    dynamic   470        F    F  nve1(1.1.1.200)
```

This example shows how to display only locally learned MAC addresses on all VLANs that are mapped to VN-Segments:

```
switch# show mac address-table local
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since first seen,+ - primary entry using vPC Peer-Link
   VLAN     MAC Address      Type      age     Secure NTFY   Ports/SWID.SSID.LID
---------+-----------------+--------+---------+------+----+------------------
```

```
* 109      0000.0410.0902    dynamic   470        F    F  Po2233
* 109      0000.0410.0912    dynamic   470        F    F  Po2233
* 108      0000.0410.0802    dynamic   470        F    F  Po2233
* 108      0000.0410.0812    dynamic   470        F    F  Po2233
* 107      0000.0410.0702    dynamic   470        F    F  Po2233
* 107      0000.0410.0712    dynamic   470        F    F  Po2233
* 106      0000.0410.0602    dynamic   470        F    F  Po2233
* 106      0000.0410.0612    dynamic   470        F    F  Po2233
* 105      0000.0410.0502    dynamic   470        F    F  Po2233
* 105      0000.0410.0512    dynamic   470        F    F  Po2233
* 104      0000.0410.0402    dynamic   470        F    F  Po2233
* 104      0000.0410.0412    dynamic   470        F    F  Po2233
* 103      0000.0410.0302    dynamic   470        F    F  Po2233
* 103      0000.0410.0312    dynamic   470        F    F  Po2233
* 102      0000.0410.0202    dynamic   470        F    F  Po2233
* 102      0000.0410.0212    dynamic   470        F    F  Po2233
* 101      0000.0410.0102    dynamic   470        F    F  Po2233
* 101      0000.0410.0112    dynamic   470        F    F  Po2233
* 100      0000.0410.0002    dynamic   470        F    F  Po2233
* 100      0000.0410.0012    dynamic   470        F    F  Po2233
switch#
```

This example shows how to display only locally learned MAC addresses on the specified VLAN, which is mapped to a VN-Segment:

```
switch# show mac address-table local vlan 107
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since first seen,+ - primary entry using vPC Peer-Link
   VLAN     MAC Address      Type      age     Secure NTFY   Ports/SWID.SSID.LID
---------+----------------+--------+---------+------+----+------------------
* 107      0000.0410.0702    dynamic   480        F    F  Po2233
* 107      0000.0410.0712    dynamic   480        F    F  Po2233
switch#
```

This example shows how to display all remote MAC addresses learned on NVE:

```
switch# show mac address-table interface nve 1
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since first seen,+ - primary entry using vPC Peer-Link
   VN_SEGMENT     MAC Address      Type       Age        Remote Vtep/VxLAN Port
--------------+----------------+---------+---------+-------------------------
* 4100            0000.0110.0002    dynamic    1180      nve1(1.1.1.200)
* 4100            0000.0110.0012    dynamic    1180      nve1(1.1.1.200)
* 4101            0000.0110.0102    dynamic    1180      nve1(1.1.1.200)
* 4101            0000.0110.0112    dynamic    1180      nve1(1.1.1.200)
* 4102            0000.0110.0202    dynamic    1180      nve1(1.1.1.200)
* 4102            0000.0110.0212    dynamic    1180      nve1(1.1.1.200)
* 4103            0000.0110.0302    dynamic    1180      nve1(1.1.1.200)
* 4103            0000.0110.0312    dynamic    1180      nve1(1.1.1.200)
* 4104            0000.0110.0402    dynamic    1180      nve1(1.1.1.200)
* 4104            0000.0110.0412    dynamic    1180      nve1(1.1.1.200)
* 4105            0000.0110.0502    dynamic    1180      nve1(1.1.1.200)
* 4105            0000.0110.0512    dynamic    1180      nve1(1.1.1.200)
* 4106            0000.0110.0602    dynamic    1180      nve1(1.1.1.200)
* 4106            0000.0110.0612    dynamic    1180      nve1(1.1.1.200)
* 4107            0000.0110.0702    dynamic    1180      nve1(1.1.1.200)
* 4107            0000.0110.0712    dynamic    1180      nve1(1.1.1.200)
* 4108            0000.0110.0802    dynamic    1180      nve1(1.1.1.200)
* 4108            0000.0110.0812    dynamic    1180      nve1(1.1.1.200)
* 4109            0000.0110.0902    dynamic    1180      nve1(1.1.1.200)
* 4109            0000.0110.0912    dynamic    1180      nve1(1.1.1.200)
switch#
```

This example shows how to display all remote MAC addresses learned on the VNI:

```
switch# show mac address-table interface nve 1 vni 4100
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since first seen,+ - primary entry using vPC Peer-Link
   VN_SEGMENT     MAC Address      Type       Age        Remote Vtep/VxLAN Port
--------------+----------------+---------+---------+-------------------------
* 4100            0000.0110.0002    dynamic    1230      nve1(1.1.1.200)
```

```
* 4100            0000.0110.0012    dynamic    1230      nve1(1.1.1.200)
switch#
```

This example shows how to display all MAC addresses learned on NVE from the specified peer:

```
switch# show mac address-table interface nve 1 peer 1.1.1.200
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since first seen,+ - primary entry using vPC Peer-Link
   VN_SEGMENT     MAC Address      Type       Age        Remote Vtep/VxLAN Port
--------------+-----------------+----------+---------+-------------------------
* 4100            0000.0110.0002    dynamic    1400      nve1(1.1.1.200)
* 4100            0000.0110.0012    dynamic    1400      nve1(1.1.1.200)
* 4101            0000.0110.0102    dynamic    1400      nve1(1.1.1.200)
* 4101            0000.0110.0112    dynamic    1400      nve1(1.1.1.200)
* 4102            0000.0110.0202    dynamic    1400      nve1(1.1.1.200)
* 4102            0000.0110.0212    dynamic    1400      nve1(1.1.1.200)
* 4103            0000.0110.0302    dynamic    1400      nve1(1.1.1.200)
* 4103            0000.0110.0312    dynamic    1400      nve1(1.1.1.200)
* 4104            0000.0110.0402    dynamic    1400      nve1(1.1.1.200)
* 4104            0000.0110.0412    dynamic    1400      nve1(1.1.1.200)
* 4105            0000.0110.0502    dynamic    1400      nve1(1.1.1.200)
* 4105            0000.0110.0512    dynamic    1400      nve1(1.1.1.200)
* 4106            0000.0110.0602    dynamic    1400      nve1(1.1.1.200)
* 4106            0000.0110.0612    dynamic    1400      nve1(1.1.1.200)
* 4107            0000.0110.0702    dynamic    1400      nve1(1.1.1.200)
* 4107            0000.0110.0712    dynamic    1400      nve1(1.1.1.200)
* 4108            0000.0110.0802    dynamic    1400      nve1(1.1.1.200)
* 4108            0000.0110.0812    dynamic    1400      nve1(1.1.1.200)
* 4109            0000.0110.0902    dynamic    1400      nve1(1.1.1.200)
* 4109            0000.0110.0912    dynamic    1400      nve1(1.1.1.200)
switch#
```

This example shows how to display all MAC addresses learned on NVE from the specified peer on the specified VNI:

```
switch# show mac address-table interface nve 1 peer 1.1.1.200 vni 4100
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since first seen,+ - primary entry using vPC Peer-Link
   VN_SEGMENT     MAC Address      Type       Age        Remote Vtep/VxLAN Port
--------------+-----------------+----------+---------+-------------------------
* 4100            0000.0110.0002    dynamic    1420      nve1(1.1.1.200)
* 4100            0000.0110.0012    dynamic    1420      nve1(1.1.1.200)
switch#
```

This example shows how to display the number of locally learned MAC address table entries:

```
switch# show mac address-table count local
MAC Entries for all vlans:
Dynamic Address Count: 20
Static Address (User-defined) Count: 0
Multicast MAC Address Count: 0
Total MAC Addresses in Use: 20

Total PVLAN Clone MAC Address Count: 0
switch#
```

This example shows how to display the number of locally learned MAC address table entries on the specified VLAN, which is mapped to a VN-Segment:

```
switch# show mac address-table count local vlan 107
MAC Entries for all vlans:
Total MAC Addresses in Use: 2
switch#
```

This example shows how to display the number of remote MAC address table entries learned on NVE:

```
switch# show mac address-table count interface nve 1
MAC entries for all vlans:
Total MAC Address in use: 20
switch#
```

This example shows how to display the number of remote MAC address table entries learned on the VNI:

```
switch# show mac address-table count interface nve 1 vni 4100
MAC entries for all vlans:
```

```
Total MAC Address in use: 2
switch#
```

This example shows how to display the number of MAC address table entries learned on NVE from the specified peer:

```
switch# show mac address-table count interface nve 1 peer 1.1.1.200
MAC entries for all vlans:
Total MAC Address in use: 20
switch#
```

This example shows how to display the number of MAC address table entries learned on NVE from the specified peer on the specified VNI:

```
switch# show mac address-table count interface nve 1 peer 1.1.1.200 vni 4100
MAC entries for all vlans:
Total MAC Address in use: 2
switch#
```

# Clearing MAC Addresses

Use one of the following commands to clear the address entries from the MAC address table:

| Command | Purpose |
|---|---|
| **clear mac address-table dynamic** | Clears all MAC address entries in the MAC address table. |
| **clear mac address-table dynamic vlan** *vlan-id* | Clears all VLAN and VXLAN MAC address entries from the MAC address table. |
| **clear mac address-table dynamic local** | Clears all locally learned MAC address entries on all VLANs mapped to VN -Segments. |
| **clear mac address-table dynamic local vlan** *vlan-id* | Clears all locally learned MAC address entries on the specified VLAN. |
| **clear mac address-table dynamic interface nve** *nve-id* | Clears all overlay learned MAC addresses. |
| **clear mac address-table dynamic interface nve** *nve-id* **vni** *vni-id* | Clears all network-learnt MAC addresses on the specified VNI. |
| **clear mac address-table dynamic interface Ethernet** *slot/port* **vlan** *vlan-id* | Clears all MAC addresses on the specified interface and VLAN. |
| **clear mac address-table dynamic interface nve** *nve-id* **peer** *ip-address*<br>**clear mac address-table dynamic interface nve** *nve-id* **peer** *ip-address* **vrf** *vrf-name* | Clears all MAC addresses on the NVE interface for the specified peer. |
| **clear mac address-table dynamic interface nve** *nve-id* **peer** *ip-address* **vni** *vni-id*<br>**clear mac address-table dynamic interface nve** *nve-id* **peer** *ip-address* **vrf** *vrf-name* **vni** *vni-id* | Clears all MAC addresses on the NVE interface from the specified peer on the specified VNI. |

This example shows how to clear all MAC address entries in the MAC address table:

```
switch# clear mac address-table dynamic
switch#
```

This example shows how to clear all VLAN and VXLAN MAC address entries from the MAC address table:

```
switch# clear mac address-table dynamic vlan 3100
switch#
```

This example shows how to clear all locally learned MAC address entries on all VLANs mapped to VN -Segments:

```
switch# clear mac address-table dynamic local
switch#
```

This example shows how to clear all locally learned MAC address entries on the specified VLAN:

```
switch# clear mac address-table dynamic local vlan 3100
switch#
```

This example shows how to clear all overlay learned MAC addresses:

```
switch# clear mac address-table dynamic interface nve 1
switch#
```

This example shows how to clear all network-learnt MAC addresses on the specified VNI:

```
switch# clear mac address-table dynamic interface nve 1 vni 5000
switch#
```

This example shows how to clear all MAC addresses on the specified interface and VLAN:

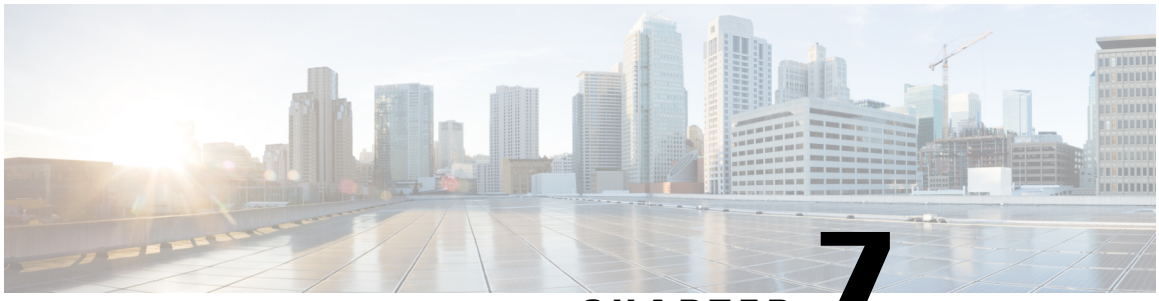
```
switch# clear mac address-table dynamic interface Ethernet 1/1 vlan 3100
switch#
```

This example shows how to clear all MAC addresses on the NVE interface for the specified peer:

```
switch# clear mac address-table dynamic interface nve 1 peer 222.1.1.1 vrf default
switch#
```

This example shows how to clear all MAC addresses on the NVE interface from the specified peer on the specified VNI:

```
switch# clear mac address-table dynamic interface nve 1 peer 222.1.1.1 vrf default vni 5000
switch#
```



CHAPTER 7

# Configuring Virtual Port Channels

This chapter contains the following sections:



# Information About vPCs

## vPC Overview

A virtual port channel (vPC) allows links that are physically connected to two different Cisco Nexus devices or Cisco Nexus Fabric Extenders to appear as a single port channel by a third device (see the following figure). The third device can be a switch, server, or any other networking device. You can configure vPCs in topologies that include Cisco Nexus devices connected to Cisco Nexus Fabric Extenders. A vPC can provide multipathing, which allows you to create redundancy by enabling multiple parallel paths between nodes and load balancing traffic where alternative paths exist.

You configure the EtherChannels by using one of the following:

- No protocol
- Link Aggregation Control Protocol (LACP)

When you configure the EtherChannels in a vPC—including the vPC peer link channel—each switch can have up to 16 active links in a single EtherChannel.

**Note** You must enable the vPC feature before you can configure or run the vPC functionality.

To enable the vPC functionality, you must create a peer-keepalive link and a peer-link under the vPC domain for the two vPC peer switches to provide the vPC functionality.

To create a vPC peer link you configure an EtherChannel on one Cisco Nexus device by using two or more Ethernet ports. On the other switch, you configure another EtherChannel again using two or more Ethernet ports. Connecting these two EtherChannels together creates a vPC peer link.

**Note** We recommend that you configure the vPC peer-link EtherChannels as trunks.

The vPC domain includes both vPC peer devices, the vPC peer-keepalive link, the vPC peer link, and all of the EtherChannels in the vPC domain connected to the downstream device. You can have only one vPC domain ID on each vPC peer device.

**Note** Always attach all vPC devices using EtherChannels to both vPC peer devices.

A vPC provides the following benefits:

- Allows a single device to use an EtherChannel across two upstream devices
- Eliminates Spanning Tree Protocol (STP) blocked ports
- Provides a loop-free topology
- Uses all available uplink bandwidth
- Provides fast convergence if either the link or a switch fails
- Provides link-level resiliency
- Assures high availability

# Terminology

## vPC Terminology

The terminology used in vPCs is as follows:

- vPC—combined EtherChannel between the vPC peer devices and the downstream device.
- vPC peer device—One of a pair of devices that are connected with the special EtherChannel known as the vPC peer link.
- vPC peer link—link used to synchronize states between the vPC peer devices.
- vPC member port—Interfaces that belong to the vPCs.

- vPC domain—domain that includes both vPC peer devices, the vPC peer-keepalive link, and all of the port channels in the vPC connected to the downstream devices. It is also associated to the configuration mode that you must use to assign vPC global parameters. The vPC domain ID must be the same on both switches.
- vPC peer-keepalive link—The peer-keepalive link monitors the vitality of a vPC peer Cisco Nexus device. The peer-keepalive link sends configurable, periodic keepalive messages between vPC peer devices.

  No data or synchronization traffic moves over the vPC peer-keepalive link; the only traffic on this link is a message that indicates that the originating switch is operating and running vPCs.

# vPC Domain

To create a vPC domain, you must first create a vPC domain ID on each vPC peer switch using a number from 1 to 1000. This ID must be the same on a set of vPC peer devices.

You can configure the EtherChannels and vPC peer links by using LACP or no protocol. When possible, we recommend that you use LACP on the peer-link, because LACP provides configuration checks against a configuration mismatch on the EtherChannel.

The vPC peer switches use the vPC domain ID that you configure to automatically assign a unique vPC system MAC address. Each vPC domain has a unique MAC address that is used as a unique identifier for the specific vPC-related operations, although the switches use the vPC system MAC addresses only for link-scope operations, such as LACP. We recommend that you create each vPC domain within the contiguous network with a unique domain ID. You can also configure a specific MAC address for the vPC domain, rather than having the Cisco NX-OS software assign the address.

The vPC peer switches use the vPC domain ID that you configure to automatically assign a unique vPC system MAC address. The switches use the vPC system MAC addresses only for link-scope operations, such as LACP or BPDUs. You can also configure a specific MAC address for the vPC domain.

We recommend that you configure the same VPC domain ID on both peers and, the domain ID should be unique in the network. For example, if there are two different VPCs (one in access and one in aggregation) then each vPC should have a unique domain ID.

After you create a vPC domain, the Cisco NX-OS software automatically creates a system priority for the vPC domain. You can also manually configure a specific system priority for the vPC domain.

**Note** If you manually configure the system priority, you must ensure that you assign the same priority value on both vPC peer switches. If the vPC peer switches have different system priority values, the vPC will not come up.

# Peer-Keepalive Link and Messages

The Cisco NX-OS software uses a peer-keepalive link between the vPC peers to transmit periodic, configurable keepalive messages. You must have Layer 3 connectivity between the peer switches to transmit these messages; the system cannot bring up the vPC peer link unless a peer-keepalive link is already up and running.

You can configure a hold-timeout and a timeout value simultaneously.

**Hold-timeout value—**The hold-timeout value range is between 3 to 10 seconds, with a default value of 3 seconds. This timer starts when the vPC peer link goes down. The purpose of the hold-timeout period is to prevent false-positive cases.

If you configure a hold-timeout value that is lower than the timeout value, then the vPC system ignores vPC peer-keepalive messages for the hold-timeout period and considers messages for the reminder of the timeout period. If no keepalive message is received for this period, the vPC secondary device takes over the role of the primary device. For example, if the hold-timeout value is 3 seconds and the timeout value is 5 seconds, for the first 3 seconds vPC keepalive messages are ignored (such as, when accommodating a supervisor failure for a few seconds after peer link failure) and keepalive messages are considered for the remaining timeout period of 2 seconds. After this period, the vPC secondary device takes over as the primary device, in case there is no keep alive message.

**Timeout value—**The timeout value range is between 3 to 20 seconds, with a default value of 5 seconds. This timer starts at the end of the hold-timeout interval. If you configure a timeout value that is lower than or equal to the hold-timeout value, then the timeout duration is initiated after the hold-timeout period. For example, if the timeout value is 3 seconds and the hold-timeout value is 5 seconds, the timeout period starts after 5 seconds.

**Note** We recommend that you configure the vPC peer-keepalive link on the Cisco Nexus device to run in the management VRF using the mgmt 0 interfaces. If you configure the default VRF, ensure that the vPC peer link is not used to carry the vPC peer-keepalive messages.

# Compatibility Parameters for vPC Peer Links

Many configuration and operational parameters must be identical on all interfaces in the vPC. After you enable the vPC feature and configure the peer link on both vPC peer switches, Cisco Fabric Services (CFS) messages provide a copy of the configuration on the local vPC peer switch configuration to the remote vPC peer switch. The system then determines whether any of the crucial configuration parameters differ on the two switches.

Enter the **show vpc consistency-parameters** command to display the configured values on all interfaces in the vPC. The displayed configurations are only those configurations that would limit the vPC peer link and vPC from coming up.

The compatibility check process for vPCs differs from the compatibility check for regular EtherChannels.

### New Type 2 Consistency Check on the vPC Port-Channels

A new type 2 consistency check has been added to validate the switchport mac learn settings on the vPC port-channels. The CLI **show vpc consistency-check vPC <vpc no.>** has been enhanced to display the local and peer values of the switchport mac-learn configuration. Because it is a type 2 check, vPC is operationally up even if there is a mismatch between the local and the peer values, but the mismatch can be displayed from the CLI output.

```
switch# sh vpc consistency-parameters vpc 1112

    Legend:
        Type 1 : vPC will be suspended in case of mismatch

Name                                Type            Local Value                Peer Value
-------------                        ----            ----------------------
-----------------------
Shut Lan                             1               No                         No
STP Port Type                        1               Default                     Default
STP Port Guard                       1               None                        None
STP MST Simulate PVST                1               Default                     Default
```

```
nve configuration                   1               nve                         nve
lag-id                              1               [(fa0,                       [(fa0,
                                                    0-23-4-ee-be-64, 8458,
0-23-4-ee-be-64, 8458,
                                                    0, 0), (8000,               0, 0),
(8000,
                                                    f4-4e-5-84-5e-3c, 457,
f4-4e-5-84-5e-3c, 457,
                                                    0, 0)]                      0, 0)]
mode                                1               active                       active
Speed                               1               10 Gb/s                     10 Gb/s
Duplex                              1               full                         full
Port Mode                           1               trunk                        trunk
Native Vlan                         1               1                           1
MTU                                 1               1500                         1500
Admin port mode                     1
Switchport MAC Learn                2               Enable                      Disable>
Newly added consistency parameter
vPC card type                       1               Empty                       Empty
Allowed VLANs                       -               311-400                      311-400
Local suspended VLANs               -               -
```

## Configuration Parameters That Must Be Identical

The configuration parameters in this section must be configured identically on both switches at either end of the vPC peer link.

**Note** You must ensure that all interfaces in the vPC have the identical operational and configuration parameters listed in this section.

Enter the **show vpc consistency-parameters** command to display the configured values on all interfaces in the vPC. The displayed configurations are only those configurations that would limit the vPC peer link and vPC from coming up.

The switch automatically checks for compatibility of these parameters on the vPC interfaces. The per-interface parameters must be consistent per interface, and the global parameters must be consistent globally.

- Port-channel mode: on, off, or active
- Link speed per channel
- Duplex mode per channel
- Trunk mode per channel:
  - Native VLAN
  - VLANs allowed on trunk
  - Tagging of native VLAN traffic
- Spanning Tree Protocol (STP) mode
- STP region configuration for Multiple Spanning Tree (MST)
- Enable or disable state per VLAN
- STP global settings:
  - Bridge Assurance setting

- Port type setting—We recommend that you set all vPC interfaces as normal ports
- Loop Guard settings

- STP interface settings:
  - Port type setting
  - Loop Guard
  - Root Guard

If any of these parameters are not enabled or defined on either switch, the vPC consistency check ignores those parameters.

**Note** To ensure that none of the vPC interfaces are in the suspend mode, enter the **show vpc brief** and **show vpc consistency-parameters** commands and check the syslog messages.

## Configuration Parameters That Should Be Identical

When any of the following parameters are not configured identically on both vPC peer switches, a misconfiguration might cause undesirable behavior in the traffic flow:

- MAC aging timers
- Static MAC entries
- VLAN interface—Each switch on the end of the vPC peer link must have a VLAN interface configured for the same VLAN on both ends and they must be in the same administrative and operational mode. Those VLANs configured on only one switch of the peer link do not pass traffic using the vPC or peer link. You must create all VLANs on both the primary and secondary vPC switches, or the VLAN will be suspended.
- Private VLAN configuration
- All ACL configurations and parameters
- Quality of service (QoS) configuration and parameters—Local parameters; global parameters must be identical
- STP interface settings:
  - BPDU Filter
  - BPDU Guard
  - Cost
  - Link type
  - Priority
  - VLANs (Rapid PVST+)

To ensure that all the configuration parameters are compatible, we recommend that you display the configurations for each vPC peer switch once you configure the vPC.

# Per-VLAN Consistency Check

Type-1 consistency checks are performed on a per-VLAN basis when spanning tree is enabled or disabled on a VLAN. VLANs that do not pass the consistency check are brought down on both the primary and secondary switches while other VLANs are not affected.

# vPC Auto-Recovery

When both vPC peer switches reload and only one switch reboots, auto-recovery allows that switch to assume the role of the primary switch and the vPC links will be allowed to come up after a predetermined period of time. The reload delay period in this scenario can range from 240 to 3600 seconds.

When vPCs are disabled on a secondary vPC switch due to a peer-link failure and then the primary vPC switch fails or is unable to forward traffic, the secondary switch reenables the vPCs. In this scenario, the vPC waits for three consecutive keepalive failures to recover the vPC links.

# vPC Peer Links

A vPC peer link is the link that is used to synchronize the states between the vPC peer devices.

**Note** You must configure the peer-keepalive link before you configure the vPC peer link or the peer link will not come up.

## vPC Peer Link Overview

You can have only two switches as vPC peers; each switch can serve as a vPC peer to only one other vPC peer. The vPC peer switches can also have non-vPC links to other switches.

To make a valid configuration, you configure an EtherChannel on each switch and then configure the vPC domain. You assign the EtherChannel on each switch as a peer link. For redundancy, we recommend that you should configure at least two dedicated ports into the EtherChannel; if one of the interfaces in the vPC peer link fails, the switch automatically falls back to use another interface in the peer link.

**Note** We recommend that you configure the EtherChannels in trunk mode.

Many operational parameters and configuration parameters must be the same in each switch connected by a vPC peer link. Because each switch is completely independent on the management plane, you must ensure that the switches are compatible on the critical parameters. vPC peer switches have separate control planes. After configuring the vPC peer link, you should display the configuration on each vPC peer switch to ensure that the configurations are compatible.

**Note** You must ensure that the two switches connected by the vPC peer link have certain identical operational and configuration parameters.

When you configure the vPC peer link, the vPC peer switches negotiate that one of the connected switches is the primary switch and the other connected switch is the secondary switch. By default, the Cisco NX-OS software uses the lowest MAC address to elect the primary switch. The software takes different actions on each switch—that is, the primary and secondary—only in certain failover conditions. If the primary switch fails, the secondary switch becomes the operational primary switch when the system recovers, and the previously primary switch is now the secondary switch.

You can also configure which of the vPC switches is the primary switch. If you want to configure the role priority again to make one vPC switch the primary switch, configure the role priority on both the primary and secondary vPC switches with the appropriate values, shut down the EtherChannel that is the vPC peer link on both switches by entering the **shutdown** command, and reenable the EtherChannel on both switches by entering the **no shutdown** command.

MAC addresses that are learned over vPC links are also synchronized between the peers.

Configuration information flows across the vPC peer links using the Cisco Fabric Services over Ethernet (CFSoE) protocol. All MAC addresses for those VLANs configured on both switches are synchronized between vPC peer switches. The software uses CFSoE for this synchronization.

If the vPC peer link fails, the software checks the status of the remote vPC peer switch using the peer-keepalive link, which is a link between vPC peer switches, to ensure that both switches are up. If the vPC peer switch is up, the secondary vPC switch disables all vPC ports on its switch. The data then forwards down the remaining active links of the EtherChannel.

The software learns of a vPC peer switch failure when the keepalive messages are not returned over the peer-keepalive link.

Use a separate link (vPC peer-keepalive link) to send configurable keepalive messages between the vPC peer switches. The keepalive messages on the vPC peer-keepalive link determines whether a failure is on the vPC peer link only or on the vPC peer switch. The keepalive messages are used only when all the links in the peer link fail.

# vPC Number

Once you have created the vPC domain ID and the vPC peer link, you can create EtherChannels to attach the downstream switch to each vPC peer switch. That is, you create one single EtherChannel on the downstream switch with half of the ports to the primary vPC peer switch and the other half of the ports to the secondary peer switch.

On each vPC peer switch, you assign the same vPC number to the EtherChannel that connects to the downstream switch. You will experience minimal traffic disruption when you are creating vPCs. To simplify the configuration, you can assign the vPC ID number for each EtherChannel to be the same as the EtherChannel itself (that is, vPC ID 10 for EtherChannel 10).

**Note** The vPC number that you assign to the EtherChannel that connects to the downstream switch from the vPC peer switch must be identical on both vPC peer switches.

# vPC Interactions with Other Features

## vPC and LACP

The Link Aggregation Control Protocol (LACP) uses the system MAC address of the vPC domain to form the LACP Aggregation Group (LAG) ID for the vPC.

You can use LACP on all the vPC EtherChannels, including those channels from the downstream switch. We recommend that you configure LACP with active mode on the interfaces on each EtherChannel on the vPC peer switches. This configuration allows you to more easily detect compatibility between switches, unidirectional links, and multihop connections, and provides dynamic reaction to run-time changes and link failures.

The vPC peer link supports 16 EtherChannel interfaces.

**Note** When you manually configure the system priority, you must ensure that you assign the same priority value on both vPC peer switches. If the vPC peer switches have different system priority values, vPC does not come up.

## vPC Peer Links and STP

When you first bring up the vPC functionality, STP reconverges. STP treats the vPC peer link as a special link and always includes the vPC peer link in the STP active topology.

We recommend that you set all the vPC peer link interfaces to the STP network port type so that Bridge Assurance is automatically enabled on all vPC peer links. We also recommend that you do not enable any of the STP enhancement features on VPC peer links.

You must configure a list of parameters to be identical on the vPC peer switches on both sides of the vPC peer link.

STP is distributed; that is, the protocol continues running on both vPC peer switches. However, the configuration on the vPC peer switch elected as the primary switch controls the STP process for the vPC interfaces on the secondary vPC peer switch.

The primary vPC switch synchronizes the STP state on the vPC secondary peer switch using Cisco Fabric Services over Ethernet (CFSoE).

The vPC manager performs a proposal/handshake agreement between the vPC peer switches that sets the primary and secondary switches and coordinates the two switches for STP. The primary vPC peer switch then controls the STP protocol for vPC interfaces on both the primary and secondary switches.

The Bridge Protocol Data Units (BPDUs) use the MAC address set for the vPC for the STP bridge ID in the designated bridge ID field. The vPC primary switch sends these BPDUs on the vPC interfaces.

**Note** Display the configuration on both sides of the vPC peer link to ensure that the settings are identical. Use the **show spanning-tree** command to display information about the vPC.

### CFSoE

The Cisco Fabric Services over Ethernet (CFSoE) is a reliable state transport mechanism that you can use to synchronize the actions of the vPC peer devices. CFSoE carries messages and packets for many features linked with vPC, such as STP and IGMP. Information is carried in CFS/CFSoE protocol data units (PDUs).

When you enable the vPC feature, the device automatically enables CFSoE, and you do not have to configure anything. CFSoE distributions for vPCs do not need the capabilities to distribute over IP or the CFS regions. You do not need to configure anything for the CFSoE feature to work correctly on vPCs.

You can use the **show mac address-table** command to display the MAC addresses that CFSoE synchronizes for the vPC peer link.

**Note** Do not enter the **no cfs eth distribute** or the **no cfs distribute** command. CFSoE must be enabled for vPC functionality. If you do enter either of these commands when vPC is enabled, the system displays an error message.

When you enter the **show cfs application** command, the output displays "Physical-eth," which shows the applications that are using CFSoE.

## Guidelines and Limitations for vPCs

vPCs have the following configuration guidelines and limitations:

- vPC is not qualified with IPv6.
- You must enable the vPC feature before you can configure vPC peer-link and vPC interfaces.
- You must configure the peer-keepalive link before the system can form the vPC peer link.
- The vPC peer-link needs to be formed using a minimum of two 10-Gigabit Ethernet interfaces.
- We recommend that you configure the same vPC domain ID on both peers and the domain ID should be unique in the network. For example, if there are two different vPCs (one in access and one in aggregation) then each vPC should have a unique domain ID.
- Only port channels can be in vPCs. A vPC can be configured on a normal port channel (switch-to-switch vPC topology) and on a port channel host interface (host interface vPC topology).
- You must configure both vPC peer switches; the configuration is not automatically synchronized between the vPC peer devices.
- Check that the necessary configuration parameters are compatible on both sides of the vPC peer link.
- You might experience minimal traffic disruption while configuring vPCs.
- You should configure all port channels in the vPC using LACP with the interfaces in active mode.
- You might experience traffic disruption when the first member of a vPC is brought up.
- OSPF over vPC and BFD with OSPF are supported on Cisco Nexus 3000 and 3100 Series switches.

  SVI limitation: When a BFD session is over SVI using virtual port-channel(vPC) peer-link, the BFD echo function is not supported. You must disable the BFD echo function for all sessions over SVI between vPC peer nodes using **no bfd echo** at the SVI configuration level.

- When a Layer 3 link is used for peer-keepalive instead of the mgmt interface, and the CPU queues are congested with control plane traffic, vPC peer-keepalive packets could be dropped. The CPU traffic includes routing protocol, ARP, Glean, and IPMC miss packets. When the peer-keepalive interface is a Layer 3 link instead of a mgmt interface, the vPC peer-keepalive packets are sent to the CPU on a low-priority queue.

  If a Layer 3 link is used for vPC peer-keepalives, configure the following ACL to prioritize the vPC peer-keepalive:

  ```
  ip access-list copp-system-acl-routingproto2
  30 permit udp any any eq 3200
  ```

  Here, 3200 is the default UDP port for keepalive packets. This ACL must match the configured UDP port in case the default port is changed.

# Enhancements for vPC

- Added support for the vPC redirect ACLs to avoid the MAC/adjacency moves between the vPC and the peer-link.
- Enabled **ip arp synchronize** to support faster convergence of Layer 3 traffic during the multichassis EtherChannel trunk (MCT) flap.

When the local vPC leg is down, all MACs and the adjacencies pointing to the vPC are moved to the peer-link. This involves the MAC table and the Route table programming that leads to the high convergence numbers. Due to this, the convergence depends on the MAC and the adjacency table scale. The vPC redirect ACLs are meant to avoid the MAC or the adjacency movement between the peer-link and the vPC Po.

Using the vPC redirect ACLs, you can redirect the traffic to the peer-link when the local vPC leg goes down. The vPC redirect ACLs are installed when the local vPC is down and the vPC redirect ACLs are removed when the local vPC comes up.

# Enabling and Disabling vPC Optimizations

A new CLI has been added on Cisco Nexus 3000 Series platform to enable or disable the vPC optimizations feature. The CLI should be enabled on both vPC peers to achieve fast-convergence. The syntax is **[no] fast-convergence**.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal** | Enters global configuration mode. |
| **Step 2** | **switch(config) # vpc domain <domain>**<br><br>**Example:**<br>switch(config) #vpc domain 100 | Configure the VPC domain number. |
| **Step 3** | **switch(config-vpc-domain)# fast-convergence** | Configure vPC fast convergence. |

# Link Scan Enhancements

The link up and down events are detected using the software link scan on all Cisco Nexus 3000 Series platforms. The default scan interval is 500ms. It means that the link up/down events are detected anywhere between 0-500 ms. As part of the current optimizations, the link scan interval has been modified from 500ms to 100ms. This ensures that the link up/down events in the hardware are detected within 100ms.

# Configuring Link Scan Interval

All connecting devices should have link scan interval set to 100 ms. Downstream devices do not have the vPC domain configuration. Therefore, a new CLI is required to set the link scan interval on such devices. The default link scan interval value is 500msec or 500000usec. The recommended interval value for fast convergence is 100000usec.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal** | Enters global configuration mode. |
| **Step 2** | **switch(config) # hardware link-scan interval ?** | Configure the link scan interval value in usec in the range <100000-1000000>. |

# Verifying the vPC Configuration

Use the following commands to display vPC configuration information:

| Command | Purpose |
|---|---|
| switch# **show feature** | Displays whether vPC is enabled or not. |
| switch# **show port-channel capacity** | Displays how many EtherChannels are configured and how many are still available on the switch. |
| switch# **show running-config vpc** | Displays running configuration information for vPCs. |
| switch# **show vpc brief** | Displays brief information on the vPCs. |
| switch# **show vpc consistency-parameters** | Displays the status of those parameters that must be consistent across all vPC interfaces. |
| switch# **show vpc peer-keepalive** | Displays information on the peer-keepalive messages. |

| Command | Purpose |
|---|---|
| switch# **show vpc role** | Displays the peer status, the role of the local switch, the vPC system MAC address and system priority, and the MAC address and priority for the local vPC switch. |
| switch# **show vpc statistics** | Displays statistics on the vPCs.<br>**Note** This command displays the vPC statistics only for the vPC peer device that you are working on. |

For information about the switch output, see the Command Reference for your Cisco Nexus Series switch.

# Viewing the Graceful Type-1 Check Status

This example shows how to display the current status of the graceful Type-1 consistency check:

```
switch# show vpc brief
Legend:
                (*) - local vPC is down, forwarding via vPC peer-link

vPC domain id                   : 10
Peer status                     : peer adjacency formed ok
vPC keep-alive status           : peer is alive
Configuration consistency status: success
Per-vlan consistency status     : success
Type-2 consistency status       : success
vPC role                        : secondary
Number of vPCs configured       : 34
Peer Gateway                    : Disabled
Dual-active excluded VLANs      : -
Graceful Consistency Check      : Enabled

vPC Peer-link status
---------------------------------------------------------------------
id   Port   Status Active vlans
--   ----   ------ --------------------------------------------------
1    Po1    up     1
```

# Viewing a Global Type-1 Inconsistency

When a global Type-1 inconsistency occurs, the vPCs on the secondary switch are brought down. The following example shows this type of inconsistency when there is a spanning-tree mode mismatch.

The example shows how to display the status of the suspended vPC VLANs on the secondary switch:

```
switch(config)# show vpc
Legend:
                (*) - local vPC is down, forwarding via vPC peer-link

vPC domain id                   : 10
Peer status                     : peer adjacency formed ok
vPC keep-alive status           : peer is alive
Configuration consistency status: failed
Per-vlan consistency status     : success
Configuration consistency reason: vPC type-1 configuration incompatible - STP
                                  Mode inconsistent
Type-2 consistency status       : success
```

```
vPC role                        : secondary
Number of vPCs configured       : 2
Peer Gateway                    : Disabled
Dual-active excluded VLANs      : -
Graceful Consistency Check      : Enabled

vPC Peer-link status
---------------------------------------------------------------------
id   Port   Status Active vlans
--   ----   ------ --------------------------------------------------
1    Po1    up     1-10

vPC status
----------------------------------------------------------------------------
id     Port        Status Consistency Reason                     Active vlans
------ ----------- ------ ----------- -------------------------- -----------
20     Po20        down*  failed      Global compat check failed -
30     Po30        down*  failed      Global compat check failed -
```

The example shows how to display the inconsistent status ( the VLANs on the primary vPC are not suspended) on the primary switch:

```
switch(config)# show vpc
Legend:
                (*) - local vPC is down, forwarding via vPC peer-link

vPC domain id                   : 10
Peer status                     : peer adjacency formed ok
vPC keep-alive status           : peer is alive
Configuration consistency status: failed
Per-vlan consistency status     : success
Configuration consistency reason: vPC type-1 configuration incompatible - STP Mo
de inconsistent
Type-2 consistency status       : success
vPC role                        : primary
Number of vPCs configured       : 2
Peer Gateway                    : Disabled
Dual-active excluded VLANs      : -
Graceful Consistency Check      : Enabled

vPC Peer-link status
---------------------------------------------------------------------
id   Port   Status Active vlans
--   ----   ------ --------------------------------------------------
1    Po1    up     1-10

vPC status
----------------------------------------------------------------------------
id     Port        Status Consistency Reason                     Active vlans
------ ----------- ------ ----------- -------------------------- -----------
20     Po20        up     failed      Global compat check failed 1-10
30     Po30        up     failed      Global compat check failed 1-10
```

# Viewing an Interface-Specific Type-1 Inconsistency

When an interface-specific Type-1 inconsistency occurs, the vPC port on the secondary switch is brought down while the primary switch vPC ports remain up.The following example shows this type of inconsistency when there is a switchport mode mismatch.

This example shows how to display the status of the suspended vPC VLAN on the secondary switch:

```
switch(config-if)# show vpc brief
Legend:
                (*) - local vPC is down, forwarding via vPC peer-link

vPC domain id                   : 10
Peer status                     : peer adjacency formed ok
vPC keep-alive status           : peer is alive
```

```
Configuration consistency status: success
Per-vlan consistency status     : success
Type-2 consistency status       : success
vPC role                        : secondary
Number of vPCs configured       : 2
Peer Gateway                    : Disabled
Dual-active excluded VLANs      : -
Graceful Consistency Check      : Enabled

vPC Peer-link status
---------------------------------------------------------------------
id   Port   Status Active vlans
--   ----   ------ -----------------------------------------------------
1    Po1    up     1

vPC status
----------------------------------------------------------------------------
id     Port        Status Consistency Reason                     Active vlans
------ ----------- ------ ----------- -------------------------- -----------
20     Po20        up     success     success                    1
30     Po30        down*  failed      Compatibility check failed -
                                        for port mode
```

This example shows how to display the inconsistent status (the VLANs on the primary vPC are not suspended) on the primary switch:

```
switch(config-if)# show vpc brief
Legend:
                (*) - local vPC is down, forwarding via vPC peer-link

vPC domain id                   : 10
Peer status                     : peer adjacency formed ok
vPC keep-alive status           : peer is alive
Configuration consistency status: success
Per-vlan consistency status     : success
Type-2 consistency status       : success
vPC role                        : primary
Number of vPCs configured       : 2
Peer Gateway                    : Disabled
Dual-active excluded VLANs      : -
Graceful Consistency Check      : Enabled

vPC Peer-link status
---------------------------------------------------------------------
id   Port   Status Active vlans
--   ----   ------ -----------------------------------------------------
1    Po1    up     1

vPC status
----------------------------------------------------------------------------
id     Port        Status Consistency Reason                     Active vlans
------ ----------- ------ ----------- -------------------------- -----------
20     Po20        up     success     success                    1
30     Po30        up     failed      Compatibility check failed 1
                                        for port mode
```

# Viewing a Per-VLAN Consistency Status

To view the per-VLAN consistency or inconsistency status, enter the **show vpc consistency-parameters vlans** command.

This example shows how to display the consistent status of the VLANs on the primary and the secondary switches.

```
switch(config-if)# show vpc brief
Legend:
                (*) - local vPC is down, forwarding via vPC peer-link
```

```
vPC domain id                     : 10
Peer status                       : peer adjacency formed ok
vPC keep-alive status             : peer is alive
Configuration consistency status: success
Per-vlan consistency status       : success
Type-2 consistency status         : success
vPC role                          : secondary
Number of vPCs configured         : 2
Peer Gateway                      : Disabled
Dual-active excluded VLANs        : -
Graceful Consistency Check        : Enabled

vPC Peer-link status
---------------------------------------------------------------------
id   Port   Status Active vlans
--   ----   ------ --------------------------------------------------
1    Po1    up     1-10

vPC status
----------------------------------------------------------------------------
id     Port        Status Consistency Reason                     Active vlans
------ ----------- ------ ----------- -------------------------- -----------
20     Po20        up     success     success                    1-10
30     Po30        up     success     success                    1-10
```

Entering **no spanning-tree vlan 5** command triggers the inconsistency on the primary and secondary VLANs:

```
switch(config)# no spanning-tree vlan 5
```

This example shows how to display the per-VLAN consistency status as Failed on the secondary switch:

```
switch(config)# show vpc brief
Legend:
                (*) - local vPC is down, forwarding via vPC peer-link

vPC domain id                     : 10
Peer status                       : peer adjacency formed ok
vPC keep-alive status             : peer is alive
Configuration consistency status: success
Per-vlan consistency status       : failed
Type-2 consistency status         : success
vPC role                          : secondary
Number of vPCs configured         : 2
Peer Gateway                      : Disabled
Dual-active excluded VLANs        : -
Graceful Consistency Check        : Enabled

vPC Peer-link status
---------------------------------------------------------------------
id   Port   Status Active vlans
--   ----   ------ --------------------------------------------------
1    Po1    up     1-4,6-10

vPC status
----------------------------------------------------------------------------
id     Port        Status Consistency Reason                     Active vlans
------ ----------- ------ ----------- -------------------------- -----------
20     Po20        up     success     success                    1-4,6-10
30     Po30        up     success     success                    1-4,6-10
```

This example shows how to display the per-VLAN consistency status as Failed on the primary switch:

```
switch(config)# show vpc brief
Legend:
                (*) - local vPC is down, forwarding via vPC peer-link

vPC domain id                     : 10
Peer status                       : peer adjacency formed ok
vPC keep-alive status             : peer is alive
Configuration consistency status: success
Per-vlan consistency status       : failed
```

```
Type-2 consistency status       : success
vPC role                        : primary
Number of vPCs configured       : 2
Peer Gateway                    : Disabled
Dual-active excluded VLANs      : -
Graceful Consistency Check      : Enabled

vPC Peer-link status
---------------------------------------------------------------------
id   Port   Status Active vlans
--   ----   ------ --------------------------------------------------
1    Po1    up     1-4,6-10

vPC status
----------------------------------------------------------------------------
id     Port        Status Consistency Reason                     Active vlans
------ ----------- ------ ----------- -------------------------- -----------
20     Po20        up     success     success                    1-4,6-10
30     Po30        up     success     success                    1-4,6-10
```

This example shows the inconsistency as STP Disabled:

```
switch(config)# show vpc consistency-parameters vlans

Name                          Type  Reason Code             Pass Vlans

-------------                 ----  ---------------------- ------------------------
STP Mode                      1     success                0-4095
STP Disabled                  1     vPC type-1             0-4,6-4095
                                    configuration
                                    incompatible - STP is
                                    enabled or disabled on
                                     some or all vlans
STP MST Region Name           1     success                0-4095
STP MST Region Revision       1     success                0-4095
STP MST Region Instance to    1     success                0-4095
 VLAN Mapping
STP Loopguard                 1     success                0-4095
STP Bridge Assurance          1     success                0-4095
STP Port Type, Edge           1     success                0-4095
BPDUFilter, Edge BPDUGuard
STP MST Simulate PVST         1     success                0-4095
Pass Vlans                    -                            0-4,6-4095
```

# vPC Default Settings

The following table lists the default settings for vPC parameters.

***Table 8: Default vPC Parameters***

| Parameters | Default |
|---|---|
| vPC system priority | 32667 |
| vPC peer-keepalive message | Disabled |
| vPC peer-keepalive interval | 1 second |
| vPC peer-keepalive timeout | 5 seconds |
| vPC peer-keepalive UDP port | 3200 |

# Configuring vPCs

## Enabling vPCs

You must enable the vPC feature before you can configure and use vPCs.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **feature vpc** | Enables vPCs on the switch. |
| **Step 3** | switch# **show feature** | (Optional)<br>Displays which features are enabled on the switch. |
| **Step 4** | switch# **copy running-config startup-config** | (Optional)<br>Copies the running configuration to the startup configuration. |

This example shows how to enable the vPC feature:

```
switch# configure terminal
switch(config)# feature vpc
```

## Disabling vPCs

You can disable the vPC feature.

**Note** When you disable the vPC feature, the Cisco Nexus device clears all the vPC configurations.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **no feature vpc** | Disables vPCs on the switch. |
| **Step 3** | switch# **show feature** | (Optional)<br>Displays which features are enabled on the switch. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | switch# **copy running-config startup-config** | (Optional)<br>Copies the running configuration to the startup configuration. |

This example shows how to disable the vPC feature:

```
switch# configure terminal
switch(config)# no feature vpc
```

# Creating a vPC Domain

You must create identical vPC domain IDs on both the vPC peer devices. This domain ID is used to automatically form the vPC system MAC address.

### Before You Begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **vpc domain** *domain-id* | Creates a vPC domain on the switch, and enters the vpc-domain configuration mode. There is no default *domain-id* ; the range is from 1 to 1000.<br>**Note** You can also use the **vpc domain** command to enter the vpc-domain configuration mode for an existing vPC domain. |
| **Step 3** | switch(config-vpc-domain)# **fast-convergence** | Enables the vPC optimizations feature. Use the **[no] fast-convergence** command to disable the vPC optimizations feature. The CLI should be enabled on both the vPC peers to achieve fast-convergence. |
| **Step 4** | switch# **show vpc brief** | (Optional)<br>Displays brief information about each vPC domain. |
| **Step 5** | switch# **copy running-config startup-config** | (Optional)<br>Copies the running configuration to the startup configuration. |

This example shows how to create a vPC domain:

```
switch# configure terminal
switch(config)# vpc domain 5
```

This example shows how to enforce the global level type-2 consistency check for the fast-convergence configuration.

```
switch# show vpc consistency-parameters global

    Legend:
        Type 1 : vPC will be suspended in case of mismatch

Name                        Type  Local Value            Peer Value
-------------               ----  ---------------------- -----------------------
Vlan to Vn-segment Map      1     No Relevant Maps       No Relevant Maps
QoS                         2     ([], [], [], [], [],   ([], [], [], [], [],
                                  [], [], [])            [], [], [])
Network QoS (MTU)           2     (1538, 0, 0, 0, 0, 0,  (1538, 0, 0, 0, 0, 0,
                                   0, 0)                  0, 0)

 ----------------------------------------------------------------------------
 ----------------------------------------------------------------------------
VTP pruning status          2     Disabled               Disabled
IGMP Snooping Group-Limit   2     8000                   8000
Fast Convergence            2     Enable                 Enable
Interface-vlan admin up     2     101-120
Interface-vlan routing      2     1,101-120              1
capability
Allowed VLANs               -     -                      -
Local suspended VLANs       -     -                      -
```

# Configuring Capabilities Checks for the Downgrade

A few strict capability checks are added to prevent the downgrade when the fast-convergence CLIs are configured.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | show system internal capability \| begin VPC_FAST_CONVERGENCE<br><br>**Example:**<br><br>Example:<br>switch# **show system internal capability \| begin VPC_FAST_CONVERGENCE**<br>1098) Service:fwm, Capability:CAP_FEATURE_N3K_VPC_FAST_CONVERGENCE<br>Registered by node:0x101 Description:vPC fast convergence feature is enabled<br>Value: Enabled<br>switch(config)# show system internal capability \| begin LINK_SCAN<br>1128) Service:bcm_usd Capability:<br>CAP_FEATURE_N3K_LINK_SCAN_INTERVAL (1127)<br>Description : Hardware link scan interval is configured<br>Value: Enabled | |
| **Step 2** | configure terminal | Enters the configuration mode. |
| **Step 3** | switch(config)# **show system internal capability \| begin LINK_SCAN** | |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`switch(config)# show system internal capability \| begin LINK_SCAN`<br>`1128) Service:bcm_usd Capability:`<br>`CAP_FEATURE_N3K_LINK_SCAN_INTERVAL (1127)`<br>`Description : Hardware link scan interval is configured`<br>`Value: Enabled` | |

# Configuring a vPC Keepalive Link and Messages

You can configure the destination IP for the peer-keepalive link that carries the keepalive messages. Optionally, you can configure other parameters for the keepalive messages.

The Cisco NX-OS software uses the peer-keepalive link between the vPC peers to transmit periodic, configurable keepalive messages. You must have Layer 3 connectivity between the peer devices to transmit these messages. The system cannot bring up the vPC peer link unless the peer-keepalive link is already up and running.

Ensure that both the source and destination IP addresses used for the peer-keepalive message are unique in your network and these IP addresses are reachable from the Virtual Routing and Forwarding (VRF) instance associated with the vPC peer-keepalive link.

**Note** We recommend that you configure a separate VRF instance and put a Layer 3 port from each vPC peer switch into that VRF instance for the vPC peer-keepalive link. Do not use the peer link itself to send vPC peer-keepalive messages.

### Before You Begin

Ensure that you have enabled the vPC feature.

You must configure the vPC peer-keepalive link before the system can form the vPC peer link.

You must configure both switches on either side of the vPC peer link.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **vpc domain** *domain-id* | Creates a vPC domain on the switch if it does not already exist, and enters the vpc-domain configuration mode. |
| **Step 3** | switch(config-vpc-domain)# **peer-keepalive destination** *ipaddress* [**hold-timeout** *secs* \| **interval** *msecs* {**timeout** *secs*} \| **precedence** | Configures the IPv4 address for the remote end of the vPC peer-keepalive link. |

| | Command or Action | Purpose |
|---|---|---|
| | {*prec-value* \| **network** \| **internet** \| **critical** \| **flash-override** \| **flash** \| **immediate priority** \| **routine**} \| **tos** {*tos-value* \| **max-reliability** \| **max-throughput** \| **min-delay** \| **min-monetary-cost** \| **normal**} \| **tos-byte** *tos-byte-value*} \| **source** *ipaddress* \| **vrf** {*name* \| **management vpc-keepalive**}] | **Note** The system does not form the vPC peer link until you configure a vPC peer-keepalive link.<br>The management ports and VRF are the defaults. |
| **Step 4** | switch(config-vpc-domain)# **vpc peer-keepalive destination** *ipaddress* **source** *ipaddress* | (Optional)<br>Configures a separate VRF instance and puts a Layer 3 port from each vPC peer device into that VRF for the vPC peer-keepalive link. |
| **Step 5** | switch# **show vpc peer-keepalive** | (Optional)<br>Displays information about the configuration for the keepalive messages. |
| **Step 6** | switch# **copy running-config startup-config** | (Optional)<br>Copies the running configuration to the startup configuration. |

This example shows how to configure the destination IP address for the vPC-peer-keepalive link:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# peer-keepalive destination 10.10.10.42
```

This example shows how to set up the peer keepalive link connection between the primary and secondary vPC device:

```
switch(config)# vpc domain 100
switch(config-vpc-domain)# peer-keepalive destination 192.168.2.2 source 192.168.2.1
Note:--------:: Management VRF will be used as the default VRF ::--------
switch(config-vpc-domain)#
```

This example shows how to create a separate VRF named vpc_keepalive for the vPC keepalive link and how to verify the new VRF:

```
vrf context vpc_keepalive
interface Ethernet1/31
  switchport access vlan 123
interface Vlan123
  vrf member vpc_keepalive
  ip address 123.1.1.2/30
  no shutdown
vpc domain 1
  peer-keepalive destination 123.1.1.1 source 123.1.1.2 vrf
vpc_keepalive

L3-NEXUS-2# show vpc peer-keepalive

vPC keep-alive status           : peer is alive
--Peer is alive for             : (154477) seconds, (908) msec
--Send status                   : Success
--Last send at                  : 2011.01.14 19:02:50 100 ms
--Sent on interface             : Vlan123
```

```
--Receive status               : Success
--Last receive at              : 2011.01.14 19:02:50 103 ms
--Received on interface        : Vlan123
--Last update from peer        : (0) seconds, (524) msec

vPC Keep-alive parameters
--Destination                  : 123.1.1.1
--Keepalive interval           : 1000 msec
--Keepalive timeout            : 5 seconds
--Keepalive hold timeout       : 3 seconds
--Keepalive vrf                : vpc_keepalive
--Keepalive udp port           : 3200
--Keepalive tos                : 192

The services provided by the switch , such as ping, ssh, telnet,
radius, are VRF aware. The VRF name need to be configured or
specified in order for the correct routing table to be used.
L3-NEXUS-2# ping 123.1.1.1 vrf vpc_keepalive
PING 123.1.1.1 (123.1.1.1): 56 data bytes
64 bytes from 123.1.1.1: icmp_seq=0 ttl=254 time=3.234 ms
64 bytes from 123.1.1.1: icmp_seq=1 ttl=254 time=4.931 ms
64 bytes from 123.1.1.1: icmp_seq=2 ttl=254 time=4.965 ms
64 bytes from 123.1.1.1: icmp_seq=3 ttl=254 time=4.971 ms
64 bytes from 123.1.1.1: icmp_seq=4 ttl=254 time=4.915 ms

--- 123.1.1.1 ping statistics ---
5 packets transmitted, 5 packets received, 0.00% packet loss
round-trip min/avg/max = 3.234/4.603/4.971 ms
```

# Creating a vPC Peer Link

You can create a vPC peer link by designating the EtherChannel that you want on each switch as the peer link for the specified vPC domain. We recommend that you configure the EtherChannels that you are designating as the vPC peer link in trunk mode and that you use two ports on separate modules on each vPC peer switch for redundancy.

### Before You Begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface port-channel** *channel-number* | Selects the EtherChannel that you want to use as the vPC peer link for this switch, and enters the interface configuration mode. |
| **Step 3** | switch(config-if)# **vpc peer-link** | Configures the selected EtherChannel as the vPC peer link, and enters the vpc-domain configuration mode. |
| **Step 4** | switch# **show vpc brief** | (Optional)<br>Displays information about each vPC, including information about the vPC peer link. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | switch# **copy running-config startup-config** | (Optional)<br>Copies the running configuration to the startup configuration. |

This example shows how to configure a vPC peer link:

```
switch# configure terminal
switch(config)# interface port-channel 20
switch(config-if)# vpc peer-link
```

# Checking the Configuration Compatibility

After you have configured the vPC peer link on both vPC peer switches, check that the configurations are consistent on all vPC interfaces.

The following QoS parameters support Type 2 consistency checks

- Network QoS—MTU and Pause
- Input Queuing —Bandwidth and Absolute Priority
- Output Queuing—Bandwidth and Absolute Priority

In the case of a Type 2 mismatch, the vPC is not suspended. Type 1 mismatches suspend the vPC.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **show vpc consistency-parameters** {**global**\|**interface port-channel***channel-number*} | Displays the status of those parameters that must be consistent across all vPC interfaces. |

This example shows how to check that the required configurations are compatible across all the vPC interfaces:

```
switch# show vpc consistency-parameters global
Legend:
         Type 1 : vPC will be suspended in case of mismatch
Name                          Type  Local Value            Peer Value

------------- ----  ---------------------- -----------------------
QoS                           2     ([], [], [], [], [],   ([], [], [], [], [],
                                     [])                    [])

Network QoS (MTU)             2     (1538, 0, 0, 0, 0, 0)  (1538, 0, 0, 0, 0, 0)
Network Qos (Pause)           2     (F, F, F, F, F, F)     (1538, 0, 0, 0, 0, 0)
Input Queuing (Bandwidth)     2     (100, 0, 0, 0, 0, 0)   (100, 0, 0, 0, 0, 0)
Input Queuing (Absolute       2     (F, F, F, F, F, F)     (100, 0, 0, 0, 0, 0)
Priority)
Output Queuing (Bandwidth)    2     (100, 0, 0, 0, 0, 0)   (100, 0, 0, 0, 0, 0)
Output Queuing (Absolute      2     (F, F, F, F, F, F)     (100, 0, 0, 0, 0, 0)
Priority)
STP Mode                      1     Rapid-PVST             Rapid-PVST
STP Disabled                  1     None                   None
```

```
STP MST Region Name         1     ""                    ""
STP MST Region Revision     1     0                     0
STP MST Region Instance to  1
  VLAN Mapping

STP Loopguard               1     Disabled              Disabled
STP Bridge Assurance        1     Enabled               Enabled
STP Port Type, Edge         1     Normal, Disabled,     Normal, Disabled,
BPDUFilter, Edge BPDUGuard        Disabled              Disabled
STP MST Simulate PVST       1     Enabled               Enabled
Allowed VLANs               -     1,624                 1
Local suspended VLANs       -     624                   -
switch#
```

# Enabling vPC Auto-Recovery

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **vpc domain** *domain-id* | Enters vpc-domain configuration mode for an existing vPC domain. |
| **Step 3** | switch(config-vpc-domain)# **auto-recovery reload-delay** *delay* | Enables the auto-recovery feature and sets the reload delay period. The default is disabled. |

This example shows how to enable the auto-recovery feature in vPC domain 10 and set the delay period for 240 seconds:

```
switch(config)# vpc domain 10
switch(config-vpc-domain)# auto-recovery reload-delay 240
Warning:
 Enables restoring of vPCs in a peer-detached state after reload, will wait for 240 seconds
 (by default) to determine if peer is un-reachable
```

This example shows how to view the status of the auto-recovery feature in vPC domain 10:

```
switch(config-vpc-domain)# show running-config vpc
!Command: show running-config vpc
!Time: Tue Dec  7 02:38:44 2010


version 5.0(3)U2(1)
feature vpc
vpc domain 10
  peer-keepalive destination 10.193.51.170
  auto-recovery
```

# Configuring the Restore Time Delay

You can configure a restore timer that delays the vPC from coming back up until after the peer adjacency forms and the VLAN interfaces are back up. This feature avoids packet drops if the routing tables fail to converge before the vPC is once again passing traffic.

### Before You Begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link with the following procedures.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **vpc domain** *domain-id* | Creates a vPC domain on the switch if it does not already exist, and enters vpc-domain configuration mode. |
| **Step 3** | switch(config-vpc-domain)# **delay restore** *time* | Configures the time delay before the vPC is restored.<br>The restore time is the number of seconds to delay bringing up the restored vPC peer device. The range is from 1 to 3600. The default is 30 seconds. |
| **Step 4** | switch# **copy running-config startup-config** | (Optional)<br>Copies the running configuration to the startup configuration. |

This example shows how to configure the delay reload time for a vPC link:

```
switch(config)# vpc domain 1
switch(config-vpc-domain)# delay restore 10
switch(config-vpc-domain)#
```

# Excluding VLAN Interfaces from Shutting Down a vPC Peer Link Fails

When a vPC peer-link is lost, the vPC secondary switch suspends its vPC member ports and its switch virtual interface (SVI) interfaces. All Layer 3 forwarding is disabled for all VLANs on the vPC secondary switch. You can exclude specific SVI interfaces so that they are not suspended.

### Before You Begin

Ensure that the VLAN interfaces have been configured.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **vpc domain** *domain-id* | Creates a vPC domain on the switch if it does not already exist, and enters vpc-domain configuration mode. |
| **Step 3** | switch(config-vpc-domain))# **dual-active exclude interface-vlan** *range* | Specifies the VLAN interfaces that should remain up when a vPC peer-link is lost. |

| | Command or Action | Purpose |
|---|---|---|
| | | range—Range of VLAN interfaces that you want to exclude from shutting down. The range is from 1 to 4094. |

This example shows how to keep the interfaces on VLAN 10 up on the vPC peer switch if a peer link fails:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# dual-active exclude interface-vlan 10
switch(config-vpc-domain)#
```

# Configuring the VRF Name

The switch services, such as ping, ssh, telnet, radius, are VRF aware. You must configure the VRF name in order for the correct routing table to be used.

You can specify the VRF name.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **ping** *ipaddress* **vrf** *vrf-name* | Specifies the virtual routing and forwarding (VRF) name to use. The VRF name is case sensitive and can be a maximum of 32 characters.. |

This example shows how to specifiy the VRF named vpc_keepalive:

```
switch# ping 123.1.1.1 vrf vpc_keepalive
PING 123.1.1.1 (123.1.1.1): 56 data bytes
64 bytes from 123.1.1.1: icmp_seq=0 ttl=254 time=3.234 ms
64 bytes from 123.1.1.1: icmp_seq=1 ttl=254 time=4.931 ms
64 bytes from 123.1.1.1: icmp_seq=2 ttl=254 time=4.965 ms
64 bytes from 123.1.1.1: icmp_seq=3 ttl=254 time=4.971 ms
64 bytes from 123.1.1.1: icmp_seq=4 ttl=254 time=4.915 ms

--- 123.1.1.1 ping statistics ---
5 packets transmitted, 5 packets received, 0.00% packet loss
round-trip min/avg/max = 3.234/4.603/4.971 ms
```

# Moving Other Port Channels into a vPC

### Before You Begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link with the following procedure.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface port-channel** *channel-number* | Selects the port channel that you want to put into the vPC to connect to the downstream switch, and enters interface configuration mode.<br>**Note** A vPC can be configured on a normal port channel (physical vPC topology) and on a port channel host interface (host interface vPC topology) |
| **Step 3** | switch(config-if)# **vpc** *number* | Configures the selected port channel into the vPC to connect to the downstream switch. The range is from 1 to 4096.<br>The vPC *number* that you assign to the port channel that connects to the downstream switch from the vPC peer switch must be identical on both vPC peer switches. |
| **Step 4** | switch# **show vpc brief** | (Optional)<br>Displays information about each vPC. |
| **Step 5** | switch# **copy running-config startup-config** | (Optional)<br>Copies the running configuration to the startup configuration. |

This example shows how to configure a port channel that will connect to the downstream device:

```
switch# configure terminal
switch(config)# interface port-channel 20
switch(config-if)# vpc 5
```

# Manually Configuring a vPC Domain MAC Address

**Note** Configuring the system address is an optional configuration step.

### Before You Begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | switch(config)# **vpc domain** *domain-id* | Selects an existing vPC domain on the switch, or creates a new vPC domain, and enters the vpc-domain configuration mode. There is no default *domain-id*; the range is from 1 to 1000. |
| **Step 3** | switch(config-vpc-domain)# **system-mac** *mac-address* | Enters the MAC address that you want for the specified vPC domain in the following format: aaaa.bbbb.cccc. |
| **Step 4** | switch# **show vpc role** | (Optional)<br>Displays the vPC system MAC address. |
| **Step 5** | switch# **copy running-config startup-config** | (Optional)<br>Copies the running configuration to the startup configuration. |

This example shows how to configure a vPC domain MAC address:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-if)# system-mac 23fb.4ab5.4c4e
```

# Manually Configuring the System Priority

When you create a vPC domain, the system automatically creates a vPC system priority. However, you can also manually configure a system priority for the vPC domain.

### Before You Begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **vpc domain** *domain-id* | Selects an existing vPC domain on the switch, or creates a new vPC domain, and enters the vpc-domain configuration mode. There is no default *domain-id*; the range is from 1 to 1000. |
| **Step 3** | switch(config-vpc-domain)# **system-priority** *priority* | Enters the system priority that you want for the specified vPC domain. The range of values is from 1 to 65535. The default value is 32667. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | switch# **show vpc brief** | (Optional)<br>Displays information about each vPC, including information about the vPC peer link. |
| **Step 5** | switch# **copy running-config startup-config** | (Optional)<br>Copies the running configuration to the startup configuration. |

This example shows how to configure a vPC peer link:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-if)# system-priority 4000
```

# Manually Configuring a vPC Peer Switch Role

By default, the Cisco NX-OS software elects a primary and secondary vPC peer switch after you configure the vPC domain and both sides of the vPC peer link. However, you may want to elect a specific vPC peer switch as the primary switch for the vPC. Then, you would manually configure the role value for the vPC peer switch that you want as the primary switch to be lower than the other vPC peer switch.

vPC does not support role preemption. If the primary vPC peer switch fails, the secondary vPC peer switch takes over to become operationally the vPC primary switch. However, the original operational roles are not restored when the formerly primary vPC comes up again.

### Before You Begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link.

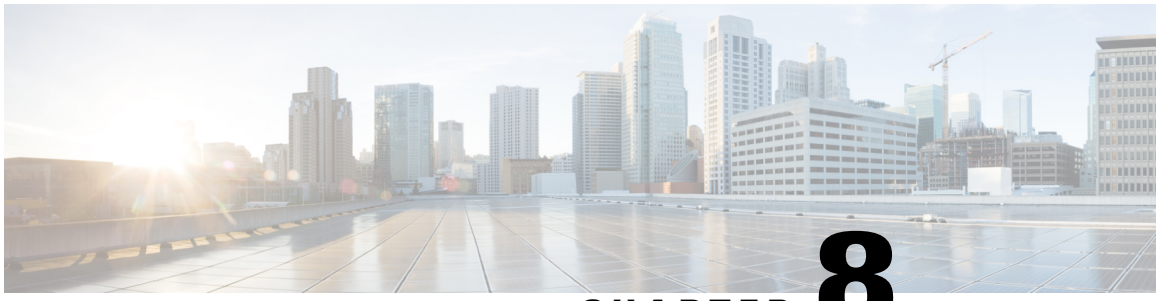
### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **vpc domain** *domain-id* | Selects an existing vPC domain on the switch, or creates a new vPC domain, and enters the vpc-domain configuration mode. There is no default *domain-id*; the range is from 1 to 1000. |
| **Step 3** | switch(config-vpc-domain)# **role priority** *priority* | Enters the role priority that you want for the vPC system priority. The range of values is from 1 to 65535. The default value is 32667. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | switch# **show vpc brief** | (Optional)<br>Displays information about each vPC, including information about the vPC peer link. |
| **Step 5** | switch# **copy running-config startup-config** | (Optional)<br>Copies the running configuration to the startup configuration. |

This example shows how to configure a vPC peer link:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-if)# role priority 4000
```



CHAPTER 8

# Configuring Q-in-Q VLAN Tunnels

This chapter contains the following sections:



## Information About Q-in-Q Tunnels

A Q-in-Q VLAN tunnel enables a service provider to segregate the traffic of different customers in their infrastructure, while still giving the customer a full range of VLANs for their internal use by adding a second 802.1Q tag to an already tagged frame.

Business customers of service providers often have specific requirements for VLAN IDs and the number of VLANs to be supported. The VLAN ranges required by different customers in the same service-provider network might overlap, and traffic of customers through the infrastructure might be mixed. Assigning a unique range of VLAN IDs to each customer would restrict customer configurations and could easily exceed the VLAN limit of 4096 of the 802.1Q specification.

**Note** Q-in-Q is supported on port channels. To configure a port channel as an asymmetrical link, all ports in the port channel must have the same tunneling configuration.

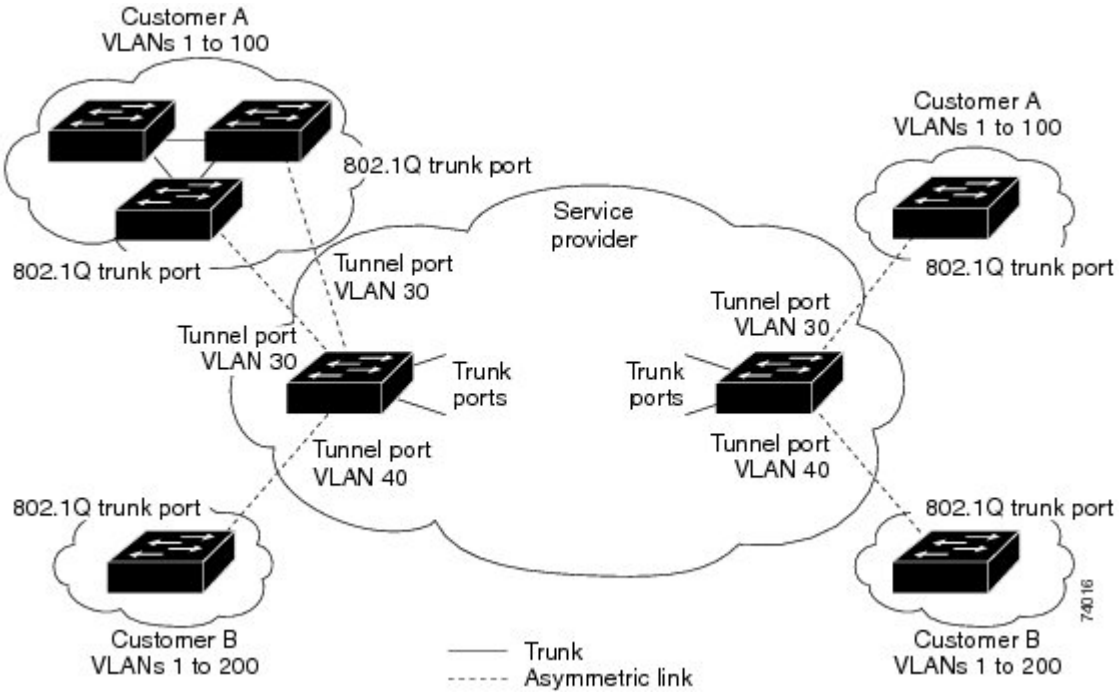
Using the 802.1Q tunneling feature, service providers can use a single VLAN to support customers who have multiple VLANs. Customer VLAN IDs are preserved and traffic from different customers is segregated within the service-provider infrastructure even when they appear to be on the same VLAN. The 802.1Q tunneling expands VLAN space by using a VLAN-in-VLAN hierarchy and tagging the tagged packets. A port configured

to support 802.1Q tunneling is called a tunnel port. When you configure tunneling, you assign a tunnel port to a VLAN that is dedicated to tunneling. Each customer requires a separate VLAN, but that VLAN supports all of the customer's VLANs.

Customer traffic tagged in the normal way with appropriate VLAN IDs come from an 802.1Q trunk port on the customer device and into a tunnel port on the service-provider edge switch. The link between the customer device and the edge switch is an asymmetric link because one end is configured as an 802.1Q trunk port and the other end is configured as a tunnel port. You assign the tunnel port interface to an access VLAN ID that is unique to each customer.

**Note** Selective Q-in-Q tunneling is not supported. All frames entering the tunnel port are subjected to Q-in-Q tagging.

***Figure 7: 802.1Q-in-Q Tunnel Ports***

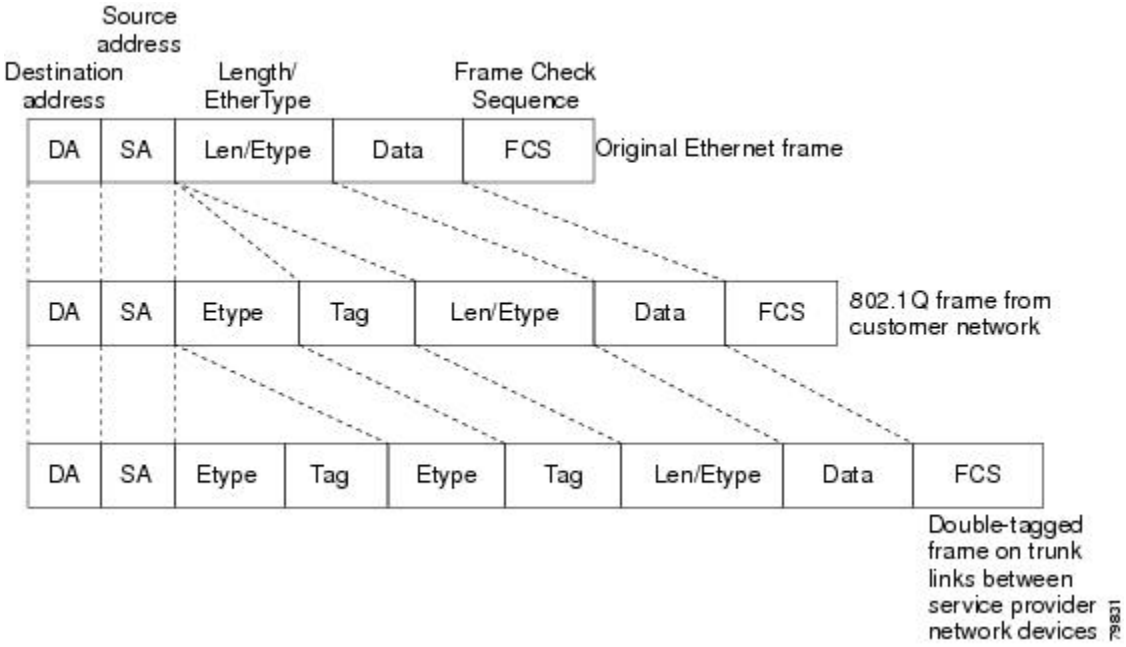



Packets that enter the tunnel port on the service-provider edge switch, which are already 802.1Q-tagged with the appropriate VLAN IDs, are encapsulated with another layer of an 802.1Q tag that contains a VLAN ID that is unique to the customer. The original 802.1Q tag from the customer is preserved in the encapsulated packet. Therefore, packets that enter the service-provider infrastructure are double-tagged.

The outer tag contains the customer's access VLAN ID (as assigned by the service provider), and the inner VLAN ID is the VLAN of the incoming traffic (as assigned by the customer). This double tagging is called tag stacking, Double-Q, or Q-in-Q.

The following figure shows the differences between the untagged, tagged and double-tagged ethernet frames.

***Figure 8: Untagged, 802.1Q-Tagged, and Double-Tagged Ethernet Frames***




By using this method, the VLAN ID space of the outer tag is independent of the VLAN ID space of the inner tag. A single outer VLAN ID can represent the entire VLAN ID space for an individual customer. This technique allows the customer's Layer 2 network to extend across the service provider network, potentially creating a virtual LAN infrastructure over multiple sites.

**Note** Hierarchical tagging, that is multi-level dot1q tagging Q-in-Q, is not supported.

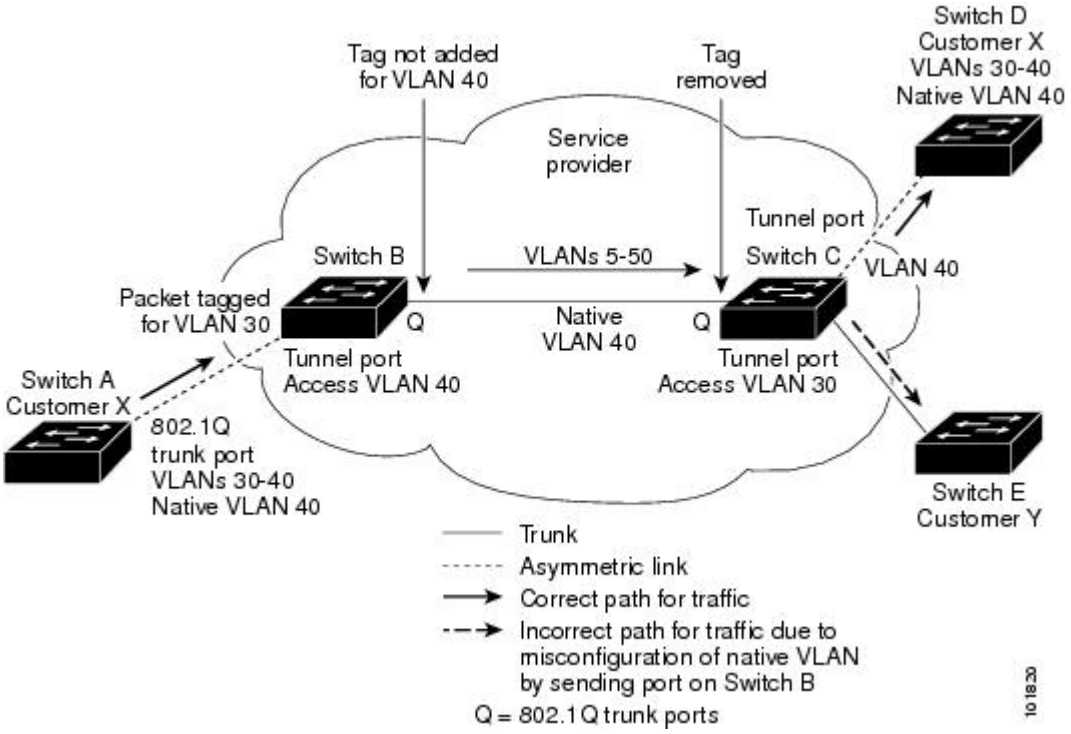
# Native VLAN Hazard

When configuring 802.1Q tunneling on an edge switch, you must use 802.1Q trunk ports for sending out packets into the service-provider network. However, packets that go through the core of the service-provider network might be carried through 802.1Q trunks, ISL trunks, or nontrunking links. When 802.1Q trunks are used in these core switches, the native VLANs of the 802.1Q trunks must not match any native VLAN of the dot1q-tunnel port on the same switch because traffic on the native VLAN is not tagged on the 802.1Q transmitting trunk port.

VLAN 40 is configured as the native VLAN for the 802.1Q trunk port from Customer X at the ingress edge switch in the service-provider network (Switch B). Switch A of Customer X sends a tagged packet on VLAN 30 to the ingress tunnel port of Switch B in the service-provider network that belongs to access VLAN 40. Because the access VLAN of the tunnel port (VLAN 40) is the same as the native VLAN of the edge-switch trunk port (VLAN 40), the 802.1Q tag is not added to the tagged packets that are received from the tunnel port. The packet carries only the VLAN 30 tag through the service-provider network to the trunk port of the egress-edge switch (Switch C) and is misdirected through the egress switch tunnel port to Customer Y.

The following figure shows the native VLAN hazard.

***Figure 9: Native VLAN Hazard***




A couple of ways to solve the native VLAN problem, are as follows:

- Configure the edge switch so that all packets going out an 802.1Q trunk, including the native VLAN, are tagged by using the **vlan dot1q tag native** command. If the switch is configured to tag native VLAN packets on all 802.1Q trunks, the switch accepts untagged packets but sends only tagged packets.

  **Note** The **vlan dot1q tag native** command is a global command that affects the tagging behavior on all trunk ports.

- Ensure that the native VLAN ID on the edge switch trunk port is not within the customer VLAN range. For example, if the trunk port carries traffic of VLANs 100 to 200, assign the native VLAN a number outside that range.

# Information About Layer 2 Protocol Tunneling

Customers at different sites connected across a service-provider network need to run various Layer 2 protocols to scale their topology to include all remote sites, as well as the local sites. The Spanning Tree Protocol (STP) must run properly, and every VLAN should build a proper spanning tree that includes the local site and all remote sites across the service-provider infrastructure. Cisco Discovery Protocol (CDP) must be able to

discover neighboring Cisco devices from local and remote sites, and the VLAN Trunking Protocol (VTP) must provide consistent VLAN configuration throughout all sites in the customer network.

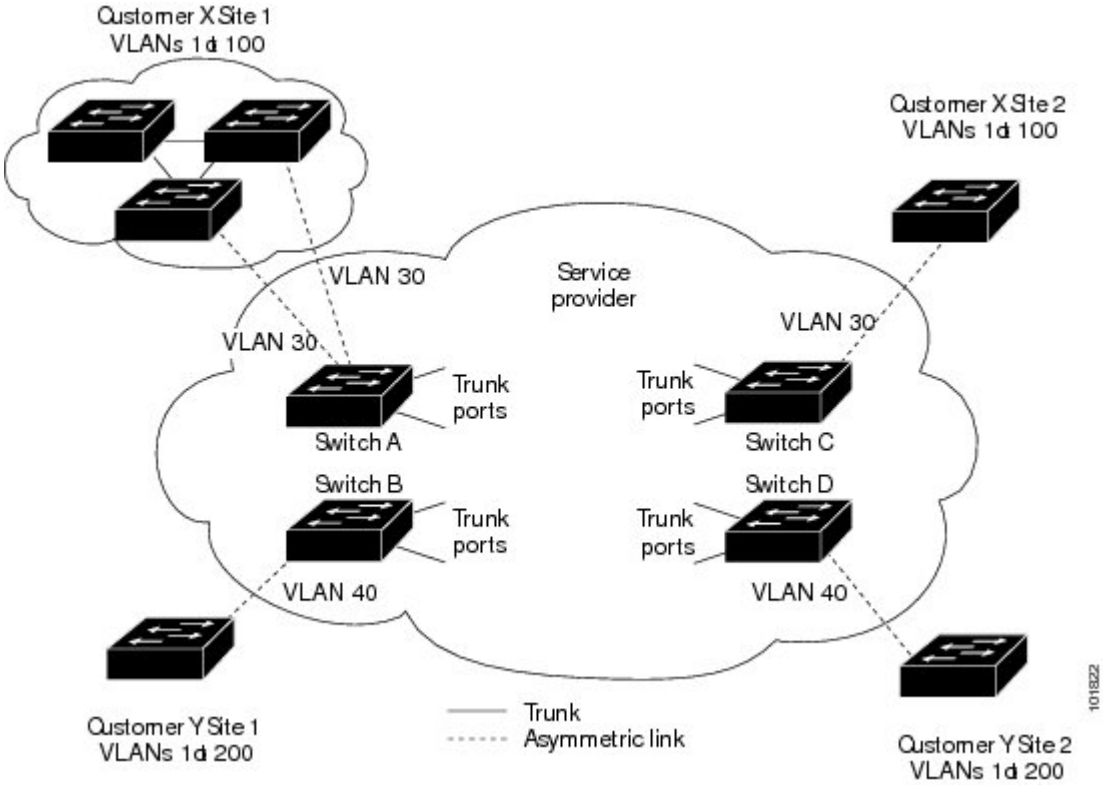
When protocol tunneling is enabled, edge switches on the inbound side of the service-provider infrastructure encapsulate Layer 2 protocol packets with a special MAC address and send them across the service-provider network. Core switches in the network do not process these packets, but forward them as normal packets. Bridge protocol data units (BPDUs) for CDP, STP, or VTP cross the service-provider infrastructure and are delivered to customer switches on the outbound side of the service-provider network. Identical packets are received by all customer ports on the same VLANs.

If protocol tunneling is not enabled on 802.1Q tunneling ports, remote switches at the receiving end of the service-provider network do not receive the BPDUs and cannot properly run STP, CDP, 802.1X, and VTP. When protocol tunneling is enabled, Layer 2 protocols within each customer's network are totally separate from those running within the service-provider network. Customer switches on different sites that send traffic through the service-provider network with 802.1Q tunneling achieve complete knowledge of the customer's VLAN.

**Note** Layer 2 protocol tunneling works by tunneling BPDUs in the software. A large number of BPDUs that comes into the supervisor module cause the CPU load to go up. The load It is controlled by Control Plane Policing CoPP configured for packets marked as BPDU.

For example, the following figure shows Customer X has four switches in the same VLAN that are connected through the service-provider network. If the network does not tunnel BPDUs, the switches on the far ends of the network cannot properly run the STP, CDP, 802.1X, and VTP protocols.

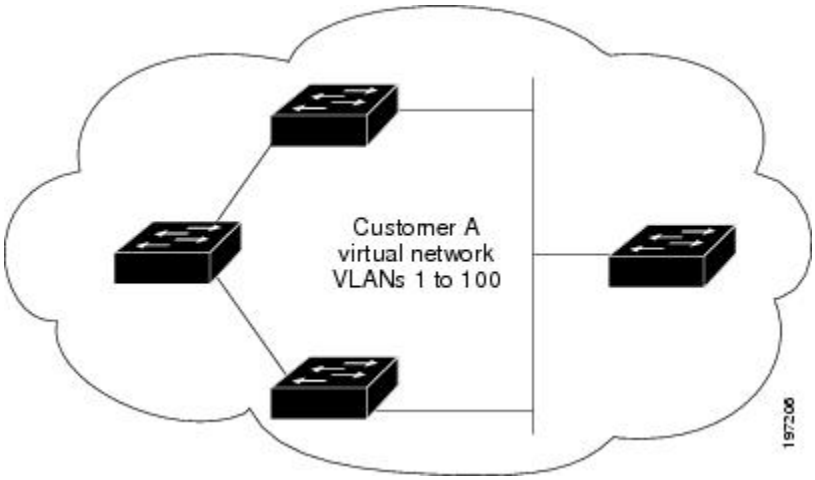
***Figure 10: Layer 2 Protocol Tunneling***




In the preceding example, STP for a VLAN on a switch in Customer X, Site 1 will build a spanning tree on the switches at that site without considering convergence parameters based on Customer X's switch in Site 2.

The following figure shows the resulting topology on the customer's network when BPDU tunneling is not enabled.

***Figure 11: Virtual Network Topology Without BPDU Tunneling***




# Licensing Requirements for Q-in-Q Tunnels

| Product | License Requirement |
|---|---|
| Cisco NX-OS | 802.1Q-in-Q VLAN tunneling and L2 protocol tunneling require no license. Any feature not included in a license package is bundled with the Cisco NX-OS system images and is provided at no extra charge to you. For a complete explanation of the Cisco NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*. |

# Guidelines and Limitations for Q-in-Q Tunneling

Q-in-Q tunnels and Layer 2 tunneling have the following configuration guidelines and limitations:

- Switches in the service-provider network must be configured to handle the increase in MTU size due to Q-in-Q tagging.
- Selective Q-in-Q tunneling is not supported. All frames that enter the tunnel port will be subject to Q-in-Q tagging.
- MAC address learning for Q-in-Q tagged packets is based on the outer VLAN (Service Provider VLAN) tag. Packet forwarding issues may occur in deployments where a single MAC address is used across multiple inner (customer) VLANs.
- Layer 3 and higher parameters cannot be identified in tunnel traffic (for example, Layer 3 destination and source addresses). Tunneled traffic cannot be routed.

- You should use MAC address-based frame distribution.
- You cannot configure the 802.1Q tunneling feature on ports that are configured to support private VLANs. Private VLAN are not required in these deployments.
- CDP must be explicitly disabled, as needed, on the dot1Q tunnel port.
- You must disable IGMP snooping on the tunnel VLANs.
- You should run the **vlan dot1Q tag native** command to maintain the tagging on the native VLAN and drop untagged traffic to prevent native VLAN misconfigurations.
- You must manually configure the 802.1Q interfaces to be edge ports.
- Dot1x tunneling is not supported.

# Configuring Q-in-Q Tunnels and Layer 2 Protocol Tunneling

## Creating a 802.1Q Tunnel Port

You create the dot1q-tunnel port using the **switchport** mode command.

**Note** You must set the 802.1Q tunnel port to an edge port with the **spanning-tree port type edge** command. The VLAN membership of the port is changed when you enter the **switchport access vlan vlan-id** command.

You should disable IGMP snooping on the access VLAN allocated for the dot1q-tunnel port to allow multicast packets to traverse the Q-in-Q tunnel.

### Before You Begin

You must first configure the interface as a switchport.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface ethernet** *slot*/*port* | Specifies an interface to configure, and enters interface configuration mode. |
| **Step 3** | switch(config-if)# **switchport** | Sets the interface as a Layer 2 switching port. |
| **Step 4** | switch(config-if)# [**no**] **switchport mode dot1q-tunnel** | Creates an 802.1Q tunnel on the port. The port will go down and reinitialize (port flap) when the interface mode is changed. BPDU filtering is enabled and CDP is disabled on tunnel interfaces. |
| **Step 5** | switch(config-if)# **exit** | Exits interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | switch(config)# **show dot1q-tunnel** [**interface** *if-range*] | (Optional)<br>Displays all ports that are in dot1q-tunnel mode. Optionally you can specify an interface or range of interfaces to display. |
| **Step 7** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to create an 802.1Q tunnel port:

```
switch# configure terminal
switch(config)# interface ethernet 7/1
switch(config-if)# switchport
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# exit
switch(config)# exit
switch# show dot1q-tunnel
```

# Enabling the Layer 2 Protocol Tunnel

You can enable protocol tunneling on the 802.1Q tunnel port.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface ethernet** *slot*/*port* | Specifies an interface to configure, and enters interface configuration mode. |
| **Step 3** | switch(config)# **switchport** | Sets the interface as a Layer 2 switching port. |
| **Step 4** | switch(config-if)# **switchport mode dot1q-tunnel** | Creates an 802.1Q tunnel on the port. |
| **Step 5** | switch(config-if)# [**no**] **l2protocol tunnel** [**cdp** \| **stp** \| **vtp**] | Enables Layer 2 protocol tunneling. Optionally, you can enable CDP, STP, or VTP tunneling. |
| **Step 6** | switch(config-if)# **exit** | Exits interface configuration mode. |
| **Step 7** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to enable protocol tunneling on an 802.1Q tunnel port:

```
switch# configure terminal
switch(config)# interface ethernet 7/1
switch(config-if)# switchport
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# l2protocol tunnel stp
switch(config-if)# exit
switch(config)# exit
```

# Configuring Thresholds for Layer 2 Protocol Tunnel Ports

You can specify the port drop and shutdown value for a Layer 2 protocol tunneling port.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **interface ethernet** *slot*/*port* | Specifies an interface to configure, and enters interface configuration mode. |
| **Step 3** | switch(config-if)# **switchport** | Sets the interface as a Layer 2 switching port. |
| **Step 4** | switch(config-if)# **switchport mode dot1q-tunnel** | Creates an 802.1Q tunnel on the port. |
| **Step 5** | switch(config-if)# [**no**] **l2protocol tunnel drop-threshold** [**cdp** \| **stp** \| **vtp**] | Specifies the maximum number of packets that can be processed on an interface before being dropped. Optionally, you can specify CDP, STP, or VTP. Valid values for the packets are from 1 to 4096.<br>The **no** form of this command resets the threshold values to 0 and disables the drop threshold. |
| **Step 6** | switch(config-if)# [**no**] **l2protocol tunnel shutdown-threshold** [**cdp** \| **stp** \| **vtp**] | Specifies the maximum number of packets that can be processed on an interface. When the number of packets is exceeded, the port is put in error-disabled state. Optionally, you can specify the Cisco Discovery Protocol (CDP), Spanning Tree Protocol (STP), or VLAN Trunking Protocol (VTP). Valid values for the packets is from 1 to 4096. |
| **Step 7** | switch(config-if)# **exit** | Exits interface configuration mode. |
| **Step 8** | switch(config)# **copy running-config startup-config** | (Optional)<br>Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

This example shows how to configure a threshold for a Layer 2 protocol tunnel port:

```
switch# configure terminal
switch(config)# interface ethernet 7/1
```

```
switch(config-if)# switchport
switch(config-if)# switchport mode dot1q-tunnel
switch(config)# l2protocol tunnel drop-threshold 3000
switch(config)# l2protocol tunnel shutdown-threshold 3000
switch(config)# exit
switch# copy running-config startup-config
```

# Verifying the Q-in-Q Configuration

Use the following command to verify the Q-in-Q tunnel and Layer 2 protocol tunneling configuration information:

| Command | Purpose |
|---|---|
| **clear l2protocol tunnel counters** [**interface** *if-range*] | Clears all the statistics counters. If no interfaces are specified, the Layer 2 protocol tunnel statistics are cleared for all interfaces. |
| **show dot1q-tunnel** [**interface** *if-range*] | Displays a range of interfaces or all interfaces that are in dot1q-tunnel mode. |
| **show l2protocol tunnel** [**interface** *if-range* \| **vlan** *vlan-id*] | Displays Layer 2 protocol tunnel information for a range of interfaces or all dot1q-tunnel interfaces that are part of a specified VLAN or all interfaces. |
| **show l2protocol tunnel summary** | Displays a summary of all ports that have Layer 2 protocol tunnel configurations. |
| **show running-config l2pt** | Displays the current Layer 2 protocol tunnel running configuration. |

# Configuration Example for Q-in-Q and Layer 2 Protocol Tunneling

This example shows a service provider switch that is configured to process Q-in-Q for traffic coming in on Ethernet 7/1. A Layer 2 protocol tunnel is enabled for STP BPDUs. The customer is allocated VLAN 10 (outer VLAN tag).

```
switch# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
switch(config)# vlan 10
switch(config-vlan)# no shutdown
switch(config-vlan)# vlan configuration 8
switch(config-vlan-config)# no ip igmp snooping
switch(config-vlan-config)# exit
switch(config-vlan)# exit
switch(config)# interface ethernet 7/1
switch(config-if)# switchport
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# switchport access vlan 10
switch(config-if)# spanning-tree port type edge
switch(config-if)# l2protocol tunnel stp
switch(config-if)# no shutdown
switch(config-if)# exit
```

```
switch(config)# exit
switch#
```

# Feature History for Q-in-Q Tunnels and Layer 2 Protocol Tunneling

*Table 9: Feature History for Q-in-Q Tunnels and Layer 2 Protocol Tunneling*

| Feature Name | Release | Feature Information |
|---|---|---|
| Q-in-Q VLAN Tunnels | 6.0(2)U1(1) | This feature was introduced. |
| L2 Protocol Tunneling | 6.0(2)U1(1) | This feature was introduced. |

# INDEX

## E



## F



## G



## I



## L

## M



## N



## P



## Q

## R



## S



## T



## U



## V