Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide, Release 10.3(x)

First Published: 2022-08-19

Last Modified: 2023-05-08

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

CONTENTS

Preface

This preface includes the following sections:

Audience

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

Document Conventions

Command descriptions use the following conventions:

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

Convention	Description
`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 includes 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 that 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.

Related Documentation for Cisco Nexus 9000 Series Switches

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

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

Documentation Feedback

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

Communications, services, and additional information

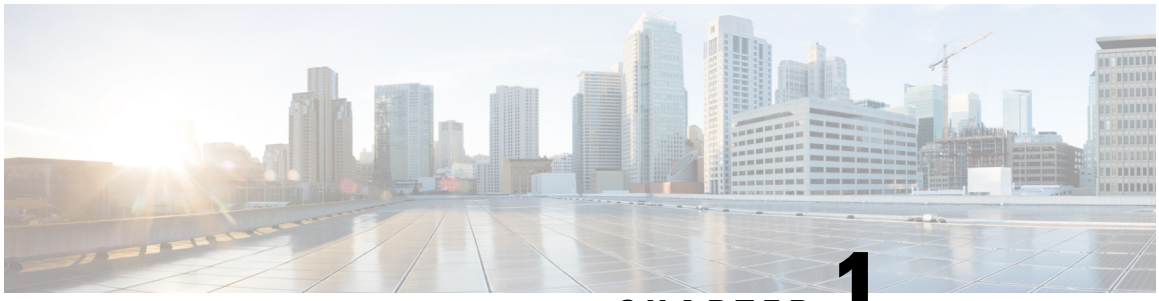
- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business impact you're looking for with the technologies that matter, visit Cisco Services.
- To submit a service request, visit Cisco Support.
- To discover and browse secure, validated enterprise-class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

Cisco Bug Search Tool

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

Documentation feedback

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



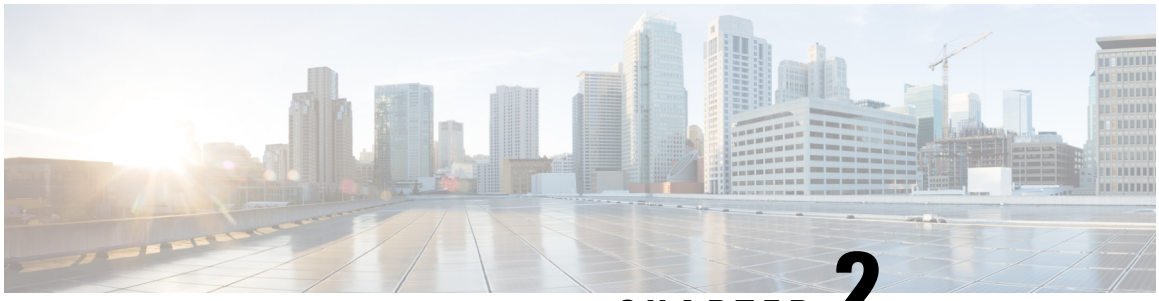
CHAPTER 1

New and Changed Information

Table 1: New and Changed Features for Release 10.3(x)

Feature	Description	Changed in Release	Where Documented
PBR for tunnel interface	Added PBR support for tunnel interface on Cisco Nexus 9300-FX2/FX3/GX/GX2 platform switches.	10.3(3)F	Point-to-Point IP-in-IP Tunnel Encapsulation and Decapsulation, on page 304 Guidelines and Limitations, on page 305
VLAN Translation on non-VXLAN topology	Added support for Port VLAN mapping on non-VXLAN VLANs.	10.3(3)F	Configuring Port VLAN Mapping on VLANs, on page 345
Hashing based on source/destination IP and L4 port number	Added hashing support of src/dst ip and src/dst L4 port number on Cisco Nexus 9808 platform switches.	10.3(1)F	Guidelines and Limitations, on page 190
Interface Consistency Checker	Added support for Interface Consistency Checker on Cisco Nexus 9800 platform switches.	10.3(1)F	Guidelines and Limitations, on page 24
Single Hop BFD	Added support for single-hop BFD on routed port, routed-sub interface, and breakout port of Cisco Nexus 9808 platform switches.	10.3(1)F	Guidelines and Limitations, on page 132
All Native and Breakout support on ports	Added native (400G, 100G, 40G) and breakout (4x100G) ports support for N9K-X9836DM-A line card of Cisco Nexus 9800 platform switches.	10.3(1)F	Guidelines and Limitations, on page 24

Feature	Description	Changed in Release	Where Documented
10G Optics support using CVR-QSFP-SFP10G adapter	Added 10G Optics support using CVR-QSFP-SFP10G adapter for N9K-X9836DM-A line card of Cisco Nexus 9808 platform switches.	10.3(1)F	Guidelines and Limitations, on page 24
Physical Interface Stats	Added statistics support for Physical Interface on Cisco Nexus 9808 platform switches.	10.3(1)F	Guidelines and Limitations, on page 24
L3 Physical and subinterface Stats	Added statistics support for L3 Physical and Subinterface on Cisco Nexus 9808 platform switches.	10.3(1)F	Guidelines and Limitations for Layer 3 Interfaces, on page 105
Hashing based	Added support for L3, Loopback, and Subinterfaces on Cisco Nexus 9808 platform switches.	10.3(1)F	Guidelines and Limitations for Layer 3 Interfaces, on page 105
UDLD	Added support for UDLD on Cisco Nexus 9808 platform switches.	10.3(1)F	Guidelines and Limitations, on page 24



CHAPTER 2

Overview

Licensing Requirements

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

Supported Platforms

Starting with Cisco NX-OS release 7.0(3)I7(1), use the Nexus Switch Platform Support Matrix to know from which Cisco NX-OS releases various Cisco Nexus 9000 and 3000 switches support a selected feature.

About Interfaces

Cisco NX-OS supports multiple configuration parameters for each of the interface types supported. Most of these parameters are covered in this guide but some are described in other documents.

The following table shows where to get further information on the parameters you can configure for an interface.

Table 2: Interface Parameters

Feature	Parameters	Further Information
Basic parameters	Description, duplex, error disable, flow control, MTU, beacon	“Configuring Basic Interface Parameters”
Layer 3	Medium, IPv4 and IPv6 addresses	“Configuring Layer 3 Interfaces”

Feature	Parameters	Further Information
Layer 3	Bandwidth, delay, IP routing, VRFs	*Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide* *Cisco Nexus 9000 Series NX-OS Multicast Routing Configuration Guide*
Port Channels	Channel group, LACP	"Configuring Port Channels"
Security	EOU	*Cisco Nexus 9000 Series NX-OS Security Configuration Guide*

Ethernet Interfaces

- Ethernet interfaces include routed ports.
- For N9K-C9316D-GX, Ports 1-16 supports 400G/100G/40G, and 10G with QSA.

Cisco Nexus N9K-C9364C-GX and N9K-C93600CD-GX Guidelines and Limitations

Cisco Nexus N9K-C9364C-GX and N9K-C93600CD-GX switches have the following guidelines and limitations:

- Consecutive groups of four interfaces (1-4, 5-8, 9-12, and so on, are referred to as a quad group). Attempting to use a mix of link speeds within a quad group is not supported. This applies to ports 1-24 of the N9K-C93600CD-GX and all ports of the N9K-C9364C-GX.
- Only one speed will be up in a quad group at a time. The first link up in a quad group determines the speed of the quad group. Ports with any other speed will be down with the reason of "Link not connected".
- The speed which remains functional when mixing within a quad group is not stored. When a mismatched speed transceiver is inserted into a quad group and brought up, all ports in the quad group will be reset. The first link which comes up after the reset determines the speed of the quad group. It is possible that pre-existing links may be shut down. You can remove the mismatched speed transceiver to recover from this state.
- FC-FEC is not supported on the second lane of the 50Gx2 breakout port. The second breakout port will not come up when 50Gx2 breakout is configured, ensure to configure RS-FEC with 50Gx2 breakout.
- Beginning with Cisco Nexus NX-OS Release 10.1(2) Auto negotiation is supported for Speed 40G and 100G on NX-OS N9K-C93600CD-GX, N9K-C9316D-GX and N9K-C9364C-GX .

Cisco Nexus N9K-X9400-16W Guidelines and Limitations

Cisco Nexus 9000 C9408 Chassis, N9K-X9400-16W (16x200G Line-Card Expansion Module (LEM)) port has the following guidelines and limitations:

- Native port supports 100G, 40G, 10G on all ports.
- Breakout ports support 4x10G, 4x25G with the following limitations:

1. The 4x10G, 4x25G breakout ports are supported only on odd ports.
2. When breakout x4 is configured on an odd port, the next corresponding even port is purged automatically.

- Breakout ports support 2x50G with the following limitations:
  1. The 2x50G breakout is supported on odd and even ports.
  2. When the 2x50G breakout is configured on an odd/even port, the corresponding even/odd port is broken out 2x50G automatically.

- 10G using QSA is supported on all ports with the following limitations:
  1. When a 10G transceiver is present on an odd/even port in the linked up state, it does not allow any other speed on the corresponding even/odd port. A warning/syslog will be printed for the mismatched XCVR and port status changes to **speed mismatch** state for the XCVR port that was inserted later. And port status will be indicated in the **show interface brief** and **show interface status** command outputs.
  2. When a 100G/40G transceiver is present on an odd/even port in linked up state, and a 10G transceiver is inserted in the corresponding even/odd port, a warning/syslog will be printed for the mismatched XCVR and port status changes to **speed mismatch** state for the XCVR port that was inserted later. And port status will be indicated in the **show interface brief** and **show interface status** command outputs.
  3. When odd port has 40G/100G and corresponding even port has 10G transceiver or vice versa, in **admin shut** status, no precedence is decided as long as the ports remain admin shut, whichever port is configured as **no shutdown** gets the first precedence.
  4. When odd port has 40G/100G and corresponding even port has 10G transceiver or vice versa, in **admin shut** status, if both ports are configured as **no shutdown** at the same time, then the port that was detected first by software gets precedence and the other gets **xcvr mismatch** state.
  5. When the state in step 4, on page 5 is achieved, in this state, if switch is reloaded, during boot-up again the port which is detected first by software takes precedence and rest gets **speed mismatch** state.

Beginning with Cisco Nexus NX-OS Release 10.5(1), the following guidelines and limitations are applicable:

- For ports 1-16, every pair of ports (1,2 | 3,4 | 5,6 | 7,8 | 9,10 | 11,12 | 13,14 | 15,16) is referred to as a quad group.
- All the ports in a quad operate in 10G with QSA, or 40G or 100G or 200G
- Mixed speed is not supported within the same quad with the following exceptions:
  - Mixed speed of 40G & 100G can be supported in quad
  - However, 100G-CR2 cannot be mixed with either 40G or other types of 100G optics in quad
- Quad speed mismatch check runs on optics insertion and removal sequence
- The first inserted transceiver in a quad group determines the speed of the quad group. Ports with any other unsupported speed is down with the reason of "**XCVR speed mismatch**. With unsupported mixed speeds, only one speed is up in a quad group at a time.

- For a particular port to be up and functional, ensure to remove all the optics or cables from all the ports in that quad and plug in the optics or cables first in the port that needs to be up and then plug in the other optics or cables.
- For a particular speed mismatch port to be up and functional, ensure to remove the optics or cables from all other ports in that quad, flap the needed port, and then plug in the other ports.
- Save the config (copy running start-up) to have port states persistency.
- When a Mismatch Transceiver is plugged into a quad, syslog is generated as **Interface Ethernet1/X is down (Reason: Inserted transceiver speed mismatch with quad speed Y).**
- Port states may not be persistent on reload ascii. Port states depends on the order of interface detected sequence on reload ascii.
- Ensure to only use the Transceivers of same speed in a quad to avoid any disruption or indeterministic state.

Access Ports

An access port carries traffic for one VLAN. This type of port is a Layer 2 interface only.

For more information on access ports, see the “Information About Access and Trunk Interfaces” section.

Routed Ports

A routed port is a physical port that can route IP traffic to another device. A routed port is a Layer 3 interface only.

For more information on routed ports, see the “Routed Interfaces” section.

Management Interface

You can use the management Ethernet interface to connect the device to a network for remote management using a Telnet client, the Simple Network Management Protocol (SNMP), or other management agents. The management port (mgmt0) is autosensing and operates in full-duplex mode at a speed of 10/100/1000 Mb/s.

For more information on the management interface, see the Cisco Nexus 9000 Series NX-OS Fundamentals Configuration Guide.

Port-Channel Interfaces

A port channel is a logical interface that is an aggregation of multiple physical interfaces. You can bundle up to 32 individual links (physical ports) into a port channel to improve bandwidth and redundancy. For more information about port-channel interfaces, see the “Configuring Port Channels” section.

Subinterfaces

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

Loopback Interfaces

A loopback interface is a virtual interface with a single endpoint that is always up. Any packet that is transmitted over a virtual loopback interface is immediately received by that interface. Loopback interfaces emulate a physical interface. For more information about subinterfaces, see the “Loopback Interfaces” section.

Breakout Interfaces

Cisco NX-OS supports the breakout of a high bandwidth interface into one or more low bandwidth interfaces at the module level or at the per-port level.

Module Level Breakout

Module Level Breakout allows certain high-density ports on a switch module to be split into multiple lower-bandwidth ports, providing increased flexibility and customization in network configurations.

You can configure the **interface breakout** command to split a high bandwidth interface of a module into multiple lower speed ports.

Some modules breakdowns the all the ports to 4x10G. 4x25G, 4x50G, 4x100G, 2x50G, 2x100G.

For example, a module level breakout 4X10G implies that high bandwidth 40G interface is broken down into four 10G interfaces. The module is reloaded and the configuration for the interface is removed when the command is executed.

The following is an example of the command:

```
switch# configure terminal
switch(config)# interface breakout module 1
Module will be reloaded. Are you sure you want to continue(yes/no)? yes
```

The **no interface breakout module** *module_number* command undoes the breakout configuration. In the above example of 4x10G interface, all interfaces of the module in 40G mode and deletes the configuration for the previous 10G interfaces.

About the Lane Selector

The lane selector is a push button switch and 4 LEDs located on the Cisco Nexus switch (left side of front panel, labeled "LS"). The push button switch and LEDs are used to indicate the status of the ports. The lane selector is supported on Cisco Nexus Series 9000 series switches and the Cisco Nexus 3164 and 3232 switches.

By default, the LEDs indicate the link/activity status of a 1 x 40G configuration. When the ports are configured as 4 x 10G, you can access the link status of each individual 10G port with the lane selector.

By pressing the lane selector push button, the port LED shows the selected lane’s link/activity status. The 1st time the push button is pressed, the first LED displays the status of the first port. Pressing the push button a 2nd time displays the status of the second port, and so on. You can display the status of each of the four ports by pressing the push button in this manner.

For example, if port 60 is configured as 4 x 10G, pressing the lane selector push button once displays the link status of 60/1/1. Pressing the push button a second time displays the link status of 60/1/2.

When you press the push button after displaying the status of the last port, all four of the LEDs should extinguish to indicate that the lane selector has returned to display the status for the default 1 x 40G configuration.

Note A 10G breakout port's LED blinks when the beacon feature has been configured for it.

Note When a port is configured to be in 10G breakout mode and no lane is selected, the 40G port's LED illuminates as green even though only one of the 10G breakout ports is up.

Support for Breakout Interfaces

The table provides detailed information of the supported or not supported breakout modes. For more information, see Cisco Nexus Data Sheets.

Table 3: Breakout Modes Support Matrix

Switches	4x10G	4x25G	2x50G	2x100G	2x200G	2x400G	4x50G	4x100G	8x100G
Nexus 9300-FX3 Platform Switches N9K-C93108TC-FX3P N9K-C93180YC-FX3	Yes	Yes	Yes	No	No	No	No	No	No
N9K-X9636C-RX	Yes	Yes	Yes	No	No	No	No	No	No
N9K-X9636C-R	Yes	Yes	Yes	No	No	No	No	No	No
N9K-X9636Q-R	Yes	No	No	No	No	No	No	No	No
N9K-X96136YC-R	No	No	No	No	No	No	No	No	No
N3K-C3636C-R	Yes	Yes	Yes	No	No	No	No	No	No
N3K-C36180YC-R	Yes	Yes	Yes	No	No	No	No	No	No
N9K-93108TC-FX3P	Yes	Yes	Yes	No	No	No	No	No	No
N9K-93108TC-EX	Yes	Yes	Yes	No	No	No	No	No	No
N9K-93180YC-EX	Yes	Yes	Yes	No	No	No	No	No	No
N9K-93108TC-FX	Yes	Yes	Yes	No	No	No	No	No	No
N9K-93180YC-FX	Yes	Yes	Yes	No	No	No	No	No	No
N9K-9348GC-FXP	Yes	Yes	Yes	No	No	No	No	No	No
N9K-X9732C-EX	Yes	Yes	Yes	No	No	No	No	No	No
N9K-X9736C-EX	Yes	Yes	Yes	No	No	No	No	No	No
N9K-X9732C-EXM	Yes	Yes	Yes	No	No	No	No	No	No

Switches	4x10G	4x25G	2x50G	2x100G	2x200G	2x400G	4x50G	4x100G	8x100G
N9K-X9736C-FX	Yes	Yes	Yes	No	No	No	No	No	No
N9K-X9736Q-FX	Yes	No	No	No	No	No	No	No	No
N9K-X9788TC-FX	Yes	Yes	Yes	No	No	No	No	No	No
N9K-X9732C-FX	Yes	Yes	Yes	No	No	No	No	No	No
N9K-C9348GC-FXP	Yes	Yes	Yes	No	No	No	No	No	No
N9K-C9336C-FX2	Yes	Yes	Yes	No	No	No	No	No	No
N9K-C93216TC-FX2	Yes	Yes	Yes	No	No	No	No	No	No
N9K-C93360YC-FX2	Yes	Yes	Yes	No	No	No	No	No	No
N9K-C9364C-GX	Yes	Yes	Yes	No	No	No	No	No	No
N9K-C9316D-GX	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No
N9K-C93600CD-GX	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No
N9K-X9716D-GX	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No
N9K-C9364D-GX2A	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No
N9K-C9332D-GX2B	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No
N9K-C9348D-GX2A	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No
N9K-X9400-16W	Yes	Yes	Yes	Yes	No	No	Yes	No	No
N9K-X9400-8D	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No
N9K-X98900CD-A	Yes	Yes	No	Yes	Yes	No	Yes	Yes	No

Guidelines and Limitations for Breakout

- Cisco Nexus 9516 switch does not support breakout on Modules 8 to 16.
- Beginning with Cisco NX-OS Release 7.0(3)F2(1), the 36-port 100-Gigabit Ethernet QSFP28 line cards (N9K-X9636C-R) and 36-port 40-Gigabit Ethernet QSFP+ line cards (N9K-X9636Q-R) support breakout of 4x10-Gigabit.
- Beginning with Cisco NX-OS Release 9.2(1), N9K-9636C-R, N9K-X9636Q-R and N9K-X9636C-RX line cards support break out of 40G ports into 4x10 Gigabits.
- Beginning with Cisco NX-OS Release 9.2(2), N9K-X9636C-R and N9K-X9636C-RX line cards support break out of 100G ports into 4x25 Gigabits. The N9K-C9636C-R does not support RS-FEC.

  Beginning with Cisco NX-OS Release 9.3(3), the default FEC mode on N9K-X9636C-R and N9K-X9636C-RX is FC-FEC for 25Gx4 and 50Gx2. When you connect N9K-X9636C-RX to N9K-X9636C-R, you must configure FC-FEC on N9K-X9636C-RX, because RS-FEC is not supported.

  The N9K-X96136YC-R line card does not support breakout.

- Beginning with Cisco NX-OS Release 9.3(3), these switches support breakout.
  - Cisco Nexus 93600CD-GX switch
  - Cisco Nexus 9500 R-Series switches support break out of 100G ports into 2x50 Gigabits.

    On Nexus 9500 R-Series switches with N9K-X9636C-R and N9K-X9636C-RX line cards, limited optics (QSFP-100G-PSM4-S, QSFP-100G-AOC, QSFP-100G-CU1M -CU3M) support 2x50G and 4x25G breakout. For more information see *Cisco Optics-to-Device Compatibility Matrix*.

Caveats

- As of Cisco NX-OS Release 7.0(3)I7(2), manual breakout of QSA ports is not supported.

  Manual breakout is supported on the following platforms because auto-breakout does not happen successfully on them—N9K-C93128TX, N9K-9332, N9K-C9396PX, N9K-C9396TX, N9K-C9372PX, N9K-C9372TX, N9K-C9332PQ, N9K-C93120TX, N9K-9432PQ, N9K-9536PQ, N9K-9636PQ, N9K-X9632PC-QSFP100, N9K-X9432C-S, N3K-C3132Q-V, N3K-C3164Q, N3K-C3132C, N3K-C3232C, N3K-C3264Q, N3K-C3264C, N3K-3064Q, N3K-3016, N3K-3172.

  You need to perform manual breakout using "interface breakout module <*module number*> port <*port range*> map <*breakout mapping*>" command.

- When a break-out port is configured as a part of a port-channel, you need to apply the configuration twice (after write-erase/reload), to ensure the effectiveness of the port-channel.

- When you upgrade a Cisco Nexus 9000 device to Cisco NX-OS release 7.0(3)I7(2) or later, if a QSFP port is configured with the manual breakout command and is using a QSA, the configuration of the interface is no longer supported and will need to be removed. To restore the configuration, you must manually configure the breakout config on the device.

  This behavior is not applicable to the following platforms—N9K-C93128TX, N9K-9332, N9K-C9396PX, N9K-C9396TX, N9K-C9372PX, N9K-C9372TX, N9K-C9332PQ, N9K-9432PQ, N9K-9536PQ, N9K-9636PQ, N9K-X9632PC-QSFP100, N9K-X9432C-S, N3K-C3132Q-V, N3K-C3164Q, N3K-C3132C, N3K-C3232C, N3K-C3264Q, N3K-C3264C, N3K-3064Q, N3K-3016, N3K-3172—because manual breakout is supported on these platforms.

- Forward error correction (FEC) is mandatory for all cable types except for 1- and 2-meter passive copper cables. The Cisco's default mode FC-FEC CL74, and also supports CONS16-RS-FEC and IEEE-RS-FEC. There are two primary FEC algorithms used in 25G Ethernet:
  - **FC-FEC** (also known as "FireCode," "BASE-R," or "Clause 74") provides low-latency error protection (under 100 nanoseconds) optimized for bursty error correction. It is used on 3- and 5-meter passive copper cables, as well as on active optical 25G cables up to 10 meters in length. This FEC type is also utilized across all 100G interfaces.
  - **RS-FEC** (also referred to as "Reed Solomon," "Clause 91," or "Clause 108") offers better error protection. It is required for 25G multimode fiber (MMF) transceivers, such as Cisco SFP-25G-SR-S, supporting distances up to 100 meters. RS-FEC may also be necessary for active optical cables exceeding 10 meters.

  All 25G devices support FC-FEC by default. The Cisco Nexus 9300-FX series is the first Cisco switch to support RS-FEC.

  Beginning with Cisco NX-OS Release 7.0(3)I7(3) you see two additional options to configure FEC such as **rs-cons16** and **rs-ieee** as per IEEE standards.

The RS FEC IEEE (25G), enabled by the **fec rs-ieee** command on Cisco Nexus 9000 switches, implements RS-FEC based for enhanced error correction on high-speed Ethernet interfaces.

```
switch# (config-if)# fec ?
auto FEC auto
fc-fec CL74(25/50G)
off Turn FEC off
rs-cons16 RS FEC Consortium 1.6 (25G)
rs-fec CL91(100G) or Consortium 1.5 (25/50G)
rs-ieee RS FEC IEEE (25G)
```

- Beginning with Cisco NX-OS Release 7.0(3)I7(7) you can display the admin and oper status of FEC interface information with the **show interface fec** command.

  Example:

```
switch# show interface fec
-------------------------------------------------------------------------------------------------
Name  Ifindex Admin-fec  Oper-fec    Status  Speed  Type
-------------------------------------------------------------------------------------------------
Eth1/1   0x1a000000 auto  auto connected     10G  SFP-H10GB-AOC2M
Eth1/2   0x1a000200      Rs-fec    notconneced    auto QSFP-100G-AOC3M
Eth1/3/1 0x38014000 auto  auto disabled auto QSFP-H40G-AOC3M
Eth1/3/2 0x38015000 auto  auto disabled auto QSFP-H40G-AOC3M
Eth1/3/3 0x38016000 auto  auto disabled auto QSFP-H40G-AOC3M
Eth1/3/4 0x38017000 auto  auto disabled auto QSFP-H40G-AOC3M
```

Note Auto-FEC is not supported in Cisco NX-OS Release 7.0(3)I7(x)

When configuring a break-out port, ensure that the FEC is matching for the link to be up.

Cisco Nexus 9000 C93180LC-EX Switch

For 7.0(3)I7(1) and later, Cisco Nexus 9000 C93180LC-EX switch provides three different modes of operation:

- Mode 1: 28 x 40G + 4 x 40G/100G (Default configuration)
  - Hardware profile portmode 4x100g + 28x40g.
  - 10x4 breakout is supported on the top ports from 1 to 27 (ports 1,3,5, 7...27). If any of the top port is broken out, the corresponding bottom port becomes non-operational. For example, if port 1 is broken out port 2 becomes non-operational.
  - 1 Gigabit and 10 Gigabit QSA is supported on ports 29, 30, 31, and 32. However, QSAs on the top and bottom front panel ports must be of same speed.
  - Ports 29, 30, 31, and 32 support 10x4, 25x4, and 50x2 breakout.
- Mode 2: 24 x 40G + 6 x 40G/100G
  - Hardware profile portmode 6x100g + 24x40g.
  - 10x4 breakout is supported on the top ports from 1 to 23 (ports 1,3,5, 7...23). If any of the top port is broken out the corresponding bottom port becomes non-operational.
  - Ports 25, 27, 29, 30, 31, and 32 support 10x4, 25x4, and 50x2 breakout.

- 1 Gigabit and 10 Gigabit QSA is supported on ports 29, 30, 31, and 32. However, QSAs on the top and bottom front panel ports must be of same speed.

- Mode 3: 18 x 40G/100G
  - Hardware profile portmode 18x100g.
  - 10x4, 25x4, and 50x2 breakout is supported on top ports from 1 to 27 (ports 1,3,5, 7...27) and on ports 29,30,31,32.
  - 1 Gigabit and 10 Gigabit QSA is supported on all the 18 ports.

Changing Mode 3 to any other mode or vice versa requires **copy running-config startup-config** command followed by **reload** command to take effect. However, moving between Modes 1 and 2 is dynamic and requires only **copy running-config startup-config** command.

Use the **show running-config | grep portmode** command to display the current operation mode.

Example:

```
switch(config-if-range)# show running-config | grep portmode

hardware profile portmode 4x100G+28x40G
```

With the Cisco Nexus C93180LC-EX switch, there are three breakout modes:

- 40G to 4x10G breakout ports
  - Enables the breakout of 40G ports into 4 X 10G ports.
  - Use the **interface breakout module 1 port** *x* **map 10g-4x** command.
- 100G to 4x25G breakout ports
  - Enables the breakout of 100G ports into 4 X 25G ports.
  - Use the **interface breakout module 1 port** *x* **map 25g-4x** command.
- 100G to 2x50G breakout ports
  - Enables the breakout of 100G ports into 2 X 50G ports.
  - Use the **interface breakout module 1 port** *x* **map 50g-2x** command.

Cisco Nexus 9000 C9364C-GX Switch

Cisco Nexus N9K-C9364C-GX breakout considerations:

- For ports 1-64, 2 x 50G, 4 x 25G and 4 x 10G breakout is supported only on odd numbered ports.
- When an odd numbered port in a quad is broken out, the even ports in that quad are removed and the other odd port in the same quad is broken out automatically to the same speed. For example, if port 1 or port 3 is broken out into 2 x 50, 4 x 25G or 4 x 10G, then the other odd port in that quad is automatically broken out to same speed and ports 2 and 4 in that quad are removed. When the above breakout configuration is removed, all ports in that quad revert to default.

- QSFP28 (100G) transceivers support the 4 x 25G breakout feature. Beginning Cisco NX-OS Release 9.3(5), the 2 x 50G breakout feature is supported.
- QSFP+ (40G) transceivers support the 4 x 10G breakout feature.
- 100G to 2x50G breakout ports
  - Enables the breakout of 100G ports into 2 X 50G ports on all odd ports.
  - Use the interface breakout module 1 port x map 50g-2x command.
- 40G to 4x10G breakout ports
  - Enables the breakout of 40G ports into 4 X 10G ports.
  - Use the interface breakout module 1 port x map 10g-4x command.

Cisco Nexus 9000 C93600CD-GX Switch

Cisco Nexus N9K-C93600CD-GX breakout considerations:

- In Cisco Nexus N9K-C93600CD-GX, every 4 ports from 1 through 24 are referred to as a quad. The breakout configuration and the speed must be same within a quad. The breakout feature may not function as expected if there is a mismatch of speed or breakout configuration within a quad. The six quads are made of ports 1-4, 5-8, 9-12, 13-16, 17-20 and 21-24.
- Beginning Cisco NX-OS Release 9.3(5), 2x50G breakout is supported on ports 1-36.
- 4x25G and 4x10G breakout is supported only on odd ports, between ports 1 through 24. The even ports will be purged within a quad (4 ports).
- When an odd-numbered port in a quad is broken out, the even ports in that quad are removed and the other odd ports within the quad is broken out automatically to the same speed. For example, if port 1 is broken out into 4x25G or 4x10G, then the other odd port in that quad is automatically broken out to same speed; and ports 2 and 4 in that quad are removed. When this breakout configuration is removed, all ports in that quad reverts to the default configuration.
- 2x50G breakout is supported on all ports from 1 through 24. All ports in a quad are broken out automatically to same speed when one port in a quad is broken out to 2x50G. For example when Port 2 is broken out into 2x50G, ports 1,3, and 4 are automatically broken out into 2x50G.

Note Only RS-FEC is supported on both lanes for 50G speed on ports 1 through 24

- Beginning with Cisco NX-OS Release 9.3(3) ports 25-28 support 4x10G, 4x25G, and 2x50G breakout features. These breakout feature are supported in port pairs. - for example 25-26 and 27-28.

Note Lane 2 of 2x50G should be configured with RS-FEC for link to be up.

- Beginning with Cisco NX-OS Release 9.3(3), consider the following breakout configuration for ports 29-36:

- QSFP-DD-400G-DR4 transceivers support only the 4 x 100G breakout feature.
- QSFP-DD-400G-FR4 and QSFP-DD-400G-LR8 transceivers do not support the breakout feature.
- QSFP28 (100G) transceivers support the 2 x 50G and 4 x 25G breakout features.
- QSFP+ (40G) transceivers support the 4 x 10G breakout feature.

Cisco Nexus 9000 C9316D-GX Switch

Cisco Nexus N9K-C9316D-GX breakout considerations:

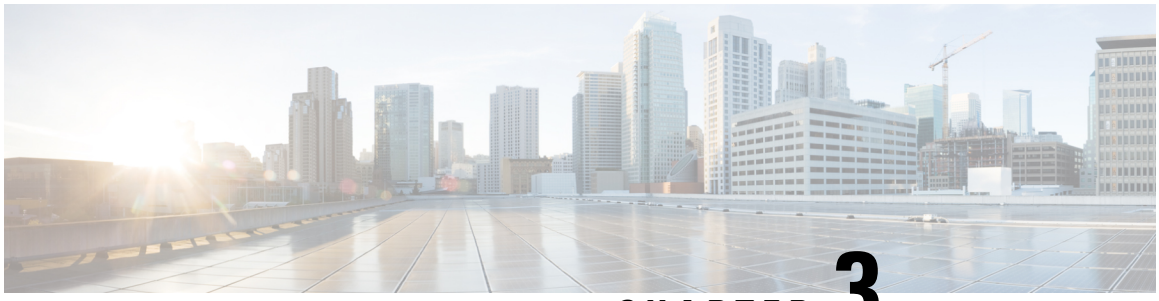
- Port 1-16 breakout consideration:
  - QSFP-DD-400G-DR4 transceivers support only the 4 x 100G and 4x10G breakout feature.
  - QSFP-DD-400G-FR4 and QSFP-DD-400G-LR8 transceivers do not support the breakout feature.
  - QSFP28 (100G) transceivers support the 2 x 50G, 4 x 25G, and 4x10G breakout feature.

Virtual Device Contexts

Cisco NX-OS can segment operating system and hardware resources into virtual device contexts (VDCs) that emulate virtual devices. The Cisco Nexus 9000 Series switch does not support multiple VDCs. All switch resources are managed in the default VDC.

High Availability for Interfaces

Interfaces support stateful and stateless restarts. A stateful restart occurs on a supervisor switchover. After the switchover, Cisco NX-OS applies the runtime configuration.



CHAPTER 3

Configuring Basic Interface Parameters

About the Basic Interface Parameters

Description

For the Ethernet and management interfaces, you can configure the description parameter to provide a recognizable name for the interface. Using a unique name for each interface allows you to quickly identify the interface when you are looking at a listing of multiple interfaces.

For information about setting the description parameter for port-channel interfaces, see the “Configuring a Port-Channel Description” section. For information about configuring this parameter for other interfaces, see the “Configuring the Description” section.

Beacon

The beacon mode allows you to identify a physical port by flashing its link state LED with a green light. By default, this mode is disabled. To identify the physical port for an interface, you can activate the beacon parameter for the interface.

For information about configuring the beacon parameter, see the “Configuring the Beacon Mode” section.

Error Disabled

A port is in the error-disabled (err-disabled) state when the port is enabled administratively (using the **no shutdown** command) but disabled at runtime by any process. For example, if UDLD detects a unidirectional link, the port is shut down at runtime. However, because the port is administratively enabled, the port status

displays as err-disable. Once a port goes into the err-disable state, you must manually reenable it or you can configure a timeout value that provides an automatic recovery. By default, the automatic recovery is not configured, and by default, the err-disable detection is enabled for all causes.

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

You can configure the automatic error-disabled recovery timeout for a particular error-disabled cause and configure the recovery period.

The **errdisable recovery cause** command provides an automatic recovery after 300 seconds.

You can use the **errdisable recovery interval** command to change the recovery period within a range of 30 to 65535 seconds. You can also configure the recovery timeout for a particular err-disable cause.

If you do not enable the error-disabled recovery for the cause, the interface stays in the error-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 error-disabled state and allowed to retry operation once all the causes have timed out.

MDIX

The medium dependent interface crossover (MDIX) parameter enables or disables the detection of a crossover connection between devices. This parameter applies only to copper interfaces. By default, this parameter is enabled. The **no mdix auto** command is supported only on N9K-C93108TC-EX, N9K-C93108TC-FX, N9K-X9788TC-FX, and N9K-C9348GC-FXP devices.

For information about configuring the MDIX parameter, see the Configuring the MDIX Parameter section.

Interface Status Error Policy

Cisco NX-OS policy servers such as Access Control List (ACL) Manager and Quality of Service (QoS) Manager, maintain a policy database. A policy is defined through the command-line interface.

Policies are pushed when you configure a policy on an interface to ensure that policies that are pushed are consistent with the hardware policies. To clear the errors and to allow the policy programming to proceed with the running configuration, enter the **no shutdown** command. If the policy programming succeeds, the port is allowed to come up. If the policy programming fails, the configuration is inconsistent with the hardware policies and the port is placed in an error-disabled policy state. The error-disabled policy state remains and the information is stored to prevent the same port from being brought up in the future. This process helps to avoid unnecessary disruption to the system.

Modifying Interface MTU Size

The maximum transmission unit (MTU) size specifies the maximum frame size that an Ethernet port can process. For transmissions to occur between two ports, you must configure the same MTU size for both ports. A port drops any frames that exceed its MTU size.

By default, the Cloud-Scale ASIC NX-OS system always allows an extra 166B in the MTU on top of the configured value in order to fully support/accept different types of encapsulations in the hardware.

Cisco NX-OS allows you to configure MTU on an interface, with options to configure it on different level in the protocol stack. By default, each interface has an MTU of 1500 bytes, which is the IEEE 802.3 standard for Ethernet frames. Larger MTU sizes are possible for more efficient processing of data to allow different application requirements. The larger frames, are also called jumbo frames, can be up to 9216 bytes in size.

MTU is configured per interface, where an interface can be a Layer 2 or a Layer 3 interface. For a Layer 2 interface, you can configure the MTU size with one of two values, the value system default MTU value or the system jumbo MTU value. The system default MTU value is 1500 bytes. Every Layer 2 interface is configured with this value by default. You can configure an interface with the default system jumbo MTU value, that is 9216 bytes. To allow an MTU value from 1500 through 9216, you must adjust the system jumbo MTU to an appropriate value where interface can be configured with the same value.

Note You can change the system jumbo MTU size. When the value is changed, the Layer 2 interfaces that use the system jumbo MTU value, will automatically changes to the new system jumbo MTU value.

A Layer 3 interface, can be Layer 3 physical interface (configure with no switchport), switch virtual interface (SVI), and sub-interface, you can configure an MTU size between 576 and 9216 bytes.

For information about setting the MTU size, see the *Configuring the MTU Size* section.

Note On Cisco Nexus 9300-FX2 and 9300-GX devices, if ingress interface is configured with an MTU less than 9216, FTE does not capture input errors and does not display any events. However, if the ingress interface is configured with an MTU of 9216, FTE displays all the events.

Bandwidth

Ethernet ports have a fixed bandwidth of 1,000,000 Kb at the physical layer. Layer 3 protocols use a bandwidth value that you can set for calculating their internal metrics. The value that you set is used for informational purposes only by the Layer 3 protocols—it does not change the fixed bandwidth at the physical layer. For example, the Enhanced Interior Gateway Routing Protocol (EIGRP) uses the minimum path bandwidth to determine a routing metric, but the bandwidth at the physical layer remains at 1,000,000 Kb.

For information about configuring the bandwidth parameter, see the Configuring the Bandwidth.

Throughput Delay

Specifying a value for the throughput-delay parameter provides a value used by Layer 3 protocols; it does not change the actual throughput delay of an interface. The Layer 3 protocols can use this value to make operating decisions. For example, the Enhanced Interior Gateway Routing Protocol (EIGRP) can use the delay setting to set a preference for one Ethernet link over another, if other parameters such as link speed are equal. The delay value that you set is in the tens of microseconds.

For information on configuring the throughput-delay parameter for other interfaces, see Configuring the Throughput Delay.

Administrative Status

The administrative-status parameter determines whether an interface is up or down. When an interface is administratively down, it is disabled and unable to transmit data. When an interface is administratively up, it is enabled and able to transmit data.

For information about configuring the administrative status parameter for port-channel interfaces, see the "Shutting Down and Restarting the Port-Channel Interface" section. For information about configuring the administrative-status parameter for other interfaces, see the "Shutting Down and Activating the Interface" section.

Unidirectional Link Detection Parameter

UDLD Overview

The Cisco-proprietary Unidirectional Link Detection (UDLD) protocol allows devices that are connected through fiber-optic 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 a device detects a unidirectional link, UDLD shuts down the affected LAN port and alerts the user. Unidirectional links can cause a variety of problems.

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 detections work to prevent physical and logical unidirectional connections and the malfunctioning of other protocols.

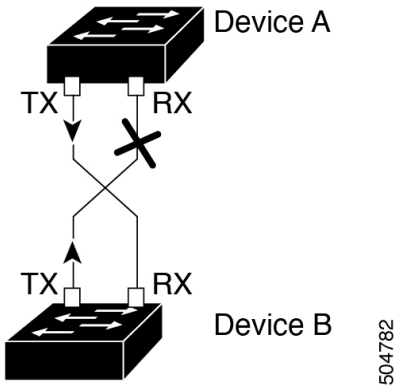
A unidirectional link occurs whenever traffic transmitted by the local device over a link is received by the neighbor but traffic transmitted from the neighbor is not received by the local device. If one of the fiber strands in a pair is disconnected, as long as autonegotiation is active, the link does 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, UDLD 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.

The Cisco Nexus 9000 Series 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. You can configure the transmission interval for the UDLD frames, either globally or for the specified interfaces.

Note By default, UDLD is locally disabled on copper LAN ports to avoid sending unnecessary control traffic on this type of media.

The 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 4: UDLD Default Configuration

Feature	Default Value
UDLD global enable state	Globally disabled
UDLD per-port enable state for fiber-optic media	Enabled on all Ethernet fiber-optic LAN ports
UDLD per-port enable state for twisted-pair (copper) media	Disabled on all Ethernet 10/100 and 1000BASE-TX LAN ports
UDLD aggressive mode	Disabled
UDLD message interval	15 seconds

For information about configuring the UDLD for the device and its port, see the “Configuring the UDLD Mode” section.

UDLD Normal and Aggressive Modes

UDLD supports Normal and Aggressive modes of operation. By default, Normal mode is enabled.

In Normal mode, UDLD detects the following link errors by examining the incoming UDLD packets from the peer port:

- Empty echo packet
- Uni-direction
- TX/RX loop
- Neighbor mismatch

By default, UDLD aggressive mode is disabled. 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 frame, UDLD tries to re-establish the connection with the neighbor. After eight failed retries, the port is disabled.

In the following scenarios, enabling the UDLD aggressive mode disables one of the ports to prevent the discarding of traffic.

- 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

Note You enable the UDLD aggressive mode globally to enable that mode on all the fiber ports. You must enable the UDLD aggressive mode on copper ports on specified interfaces.

Tip When a line card upgrade is being performed during an in-service software upgrade (ISSU) and some of the ports on the line card are members of a Layer 2 port channel and are configured with UDLD aggressive mode, if you shut down one of the remote ports, UDLD puts the corresponding port on the local device into an error-disabled state. This behavior is correct.

To restore service after the ISSU has completed, enter the **shutdown** command followed by the **no shutdown** command on the local port.

Port-Channel Parameters

A port channel is an aggregation of physical interfaces that comprise a logical interface. You can bundle up to 32 individual interfaces into a port channel to provide increased bandwidth and redundancy. Port channeling also load balances traffic across these physical interfaces. The port channel stays operational if at least one physical interface within the port channel is operational.

You can create Layer 3 port channels by bundling compatible Layer 3 interfaces.

Any configuration changes that you apply to the port channel are applied to each interface member of that port channel.

For information about port channels and for information about configuring port channels, see Chapter 6, “Configuring Port Channels.”

Port Profiles

On Cisco Nexus 9300 Series switches, you can create a port profile that contains many interface commands and apply that port profile to a range of interfaces. Each port profile can be applied only to a specific type of interface; the choices are as follows:

- Ethernet
- VLAN network interface
- Port channel

When you choose Ethernet or port channel as the interface type, the port profile is in the default mode which is Layer 3. Enter the **switchport** command to change the port profile to Layer 2 mode.

You inherit the port profile when you attach the port profile to an interface or range of interfaces. When you attach, or inherit, a port profile to an interface or range of interfaces, the system applies all the commands in that port profile to the interfaces. Additionally, you can have one port profile inherit the settings from another port profile. Inheriting another port profile allows the initial port profile to assume all of the commands of the second, inherited, port profile that do not conflict with the initial port profile. Four levels of inheritance are supported. The same port profile can be inherited by any number of port profiles.

The system applies the commands inherited by the interface or range of interfaces according to the following guidelines:

- Commands that you enter under the interface mode take precedence over the port profile's commands if there is a conflict. However, the port profile retains that command in the port profile.
- The port profile's commands take precedence over the default commands on the interface, unless the port-profile command is explicitly overridden by the default command.
- When a range of interfaces inherits a second port profile, the commands of the initial port profile override the commands of the second port profile if there is a conflict.
- After you inherit a port profile onto an interface or range of interfaces, you can override individual configuration values by entering the new value at the interface configuration level. If you remove the individual configuration values at the interface configuration level, the interface uses the values in the port profile again.
- There are no default configurations associated with a port profile.

A subset of commands are available under the port-profile configuration mode, depending on which interface type you specify.

Note You cannot use port profiles with Session Manager. See the *Cisco Nexus 9000 Series NX-OS System Management Configuration Guide* for information about Session Manager.

To apply the port-profile configurations to the interfaces, you must enable the specific port profile. You can configure and inherit a port profile onto a range of interfaces prior to enabling the port profile. You would then enable that port profile for the configurations to take effect on the specified interfaces.

If you inherit one or more port profiles onto an original port profile, only the last inherited port profile must be enabled; the system assumes that the underlying port profiles are enabled.

When you remove a port profile from a range of interfaces, the system undoes the configuration from the interfaces first and then removes the port-profile link itself. Also, when you remove a port profile, the system checks the interface configuration and either skips the port-profile commands that have been overridden by directly entered interface commands or returns the command to the default value.

If you want to delete a port profile that has been inherited by other port profiles, you must remove the inheritance before you can delete the port profile.

You can also choose a subset of interfaces from which to remove a port profile from among that group of interfaces that you originally applied the profile. For example, if you configured a port profile and configured ten interfaces to inherit that port profile, you can remove the port profile from just some of the specified ten interfaces. The port profile continues to operate on the remaining interfaces to which it is applied.

If you delete a specific configuration for a specified range of interfaces using the interface configuration mode, that configuration is also deleted from the port profile for that range of interfaces only. For example, if you have a channel group inside a port profile and you are in the interface configuration mode and you delete that port channel, the specified port channel is also deleted from the port profile as well.

Just as in the device, you can enter a configuration for an object in port profiles without that object being applied to interfaces yet. For example, you can configure a virtual routing and forward (VRF) instance without it being applied to the system. If you then delete that VRF and related configurations from the port profile, the system is unaffected.

After you inherit a port profile on an interface or range of interfaces and you delete a specific configuration value, that port-profile configuration is not operative on the specified interfaces.

If you attempt to apply a port profile to the wrong type of interface, the system returns an error.

When you attempt to enable, inherit, or modify a port profile, the system creates a checkpoint. If the port-profile configuration fails, the system rolls back to the prior configuration and returns an error. A port profile is never only partially applied.

Cisco QSFP+ to SFP+ Adapter Module Support

The Cisco QSFP+ to SFP+ Adapter (QSA) module provides 10G support for the 40G uplink ports that are a part of the Cisco Nexus M6PQ and Cisco Nexus M12PQ uplink modules of specific Cisco Nexus 9300 devices.

A group of six consecutive ports in the M6PQ or M12PQ uplink module must be operating at the same speed (40G or 10G) to use the QSA/QSFP modules.

- For Cisco Nexus 9396PX devices, 2/1-6 ports form the first port speed group and the remaining 2/7-12 ports form the second port speed group.
- For Cisco Nexus 93128PX/TX devices, 2/1-6 ports form the first port speed group and the remaining 2/7-8 ports form the second port speed group.
- For Cisco Nexus 937xPX/TX devices, 1/49-54 ports form the only port speed group.
- For Cisco Nexus 93120TX devices, 1/97-102 ports form the only port speed group.
- For Cisco Nexus 9332PQ devices, 1/27-32 ports form the only port speed group.

Use the **speed-group 10000** command to configure the first port of a port speed group for the QSA. This command specifies the administrator speed preference for the port group. (The default port speed is 40G.)

- The **speed-group 10000** command specifies a speed of 10G.
- The **no speed-group 10000** command specifies a speed of 40G.
- Uplink modules should not be removed from a Cisco Nexus 9300 platform switch that is running Cisco NX-OS Release 7.0(3)I7(5). The ports on uplink modules should be used only for uplinks.
- Beginning with Cisco NX-OS Release 9.2(2), CWDM4 is supported on the 36-port 100-Gigabit Ethernet QSFP28 line cards (N9K-X9636C-R), the 36-port 40-Gigabit Ethernet QSFP+ line cards (N9K-X9636Q-R), the 36-port 100-Gigabit QSFP28 line cards (N9K-X9636C-RX) and the 52-port 100-Gigabit QSFP28 line cards (N9K-X96136YC-R).

After the speed has been configured, the compatible transceiver modules are enabled. The remaining transceiver modules in the port group (incompatible transceiver modules) become error disabled with a reason of "check speed-group config".

Note The Cisco QSFP+ to SFP+ Adapter (QSA) module does not provide 10G support for the 40G line cards for Cisco Nexus 9500 devices.

You can use a QSFP-to-SFP adapter on Cisco Nexus 9200 and 9300-EX Series switches and Cisco Nexus 3232C and 3264Q Series switches.

Cisco SFP+ Adapter Module Support

You can use the CVR-2QSFP28-8SFP adapter for 25-Gigabit optics support on 100-Gigabit ports of the Cisco Nexus 9236C switch.

The **interface breakout module** command can be used to split this switch's 100G interfaces into four 25G interfaces. After you enter this command, you must copy the running configuration to the startup configuration.

Beginning with Cisco NX-OS Release 9.2(3), 10/25 LR is supported on N9K-C93180YC-EX, N9K-X97160YC-EX, N9K-C93180YC-FX, N9K-C93240YC-FX2 and N3K-C34180YC switches. This dual speed optical transceiver operates at 25G by default and it seamlessly interoperates with other 25G LR transceivers. Because auto speed sensing is not supported on this device, to interoperate with a 10G transceiver, you must manually configure it to use 10G speed.

Cisco SFP-10G-T-X Module Support

Beginning with Cisco NX-OS Release 9.3(5), 10G BASE-T SFP+ (RJ-45) is supported on N9K-C93240YC-FX2, N9K-C93180YC-EX, N9K-C93180YC-FX and N9K-C93360YC-FX2 devices. This copper transceiver operates at 10G by default.

Note When you connect a SFP-10G-T-X device into a port, all the neighboring ports of this device must be either empty, or be connected to passive copper links only.

Note On Cisco Nexus 9000 Series Switches, the `show interface` and `show interface capability` commands may display 100 Mbps as a supported speed for certain ports. However, this speed is only supported when using the SFP-10G-T-X transceiver. For ports using GLC-TE transceivers, the lowest supported speed is 1 Gbps.

Note Interface configured with media-type 10G-TX while in admin up state will remain errdisabled under Unsupported media-type. To remove this condition, use the following commands on the interface:

- **shutdown**
- **no shutdown**

Table 5: Default Port Mapping

Device Name	Port Map
Cisco Nexus N9K-C93180YC-EX, N9K-C93180YC-FX, N9K-C93180YC-FX3 and N9K-C93180YC-FX3S	PI/PE: 1, 4-5, 8-9, 12-13, 16, 37, 40-41, 44-45, 48
Cisco Nexus N9K-C93240YC-FX2	W/ PI Fan/PS: 2, 6, 8, 12, 14, 18, 20, 24, 26, 30, 32, 36, 38, 42, 44, 48 W/ PE Fan/PS: 6, 12, 18, 24, 30, 36, 42, 48
Cisco Nexus N9K-C93360YC-FX2	PI/PE 1, 4-5, 8, 41, 44-45, 48-49, 52-53, 56-57, 60-61, 64-65, 68-69, 72-73, 76-77, 80-81, 84-85, 88-89, 92-93, 96

Guidelines and Limitations

Basic interface parameters have the following configuration guidelines and limitations:

- MDIX is enabled by default on copper ports. It is not possible to disable it.
- **show** commands with the **internal** keyword are not supported.
- Fiber-optic Ethernet ports must use Cisco-supported transceivers. To verify that the ports are using Cisco-supported transceivers, use the **show interface transceivers** command. Interfaces with Cisco-supported transceivers are listed as functional interfaces.
- A port can be either a Layer 2 or a Layer 3 interface; it cannot be both simultaneously.

  By default, each port is a Layer 3 interface.

  You can change a Layer 3 interface into a Layer 2 interface by using the **switchport** command. You can change a Layer 2 interface into a Layer 3 interface by using the **no switchport** command.
- Flow control using pause frames is not supported.
- Beginning with Cisco NX-OS Release 9.3(1), only MTU 9216 can be configured on FEX fabric ports. Trying to configure any other value generates an error.

  If the MTU value on a FEX fabric port-channel was set to 9216 before the switch was upgraded to Cisco NX-OS Release 9.3(1), the **show running config** command does not display the MTU value, but the **show running-config diff** command does.
- Beginning with Cisco NX-OS Release 9.3(1), FEX fabric port-channels support only MTU 9216 by default.
- The following line cards do not support Link Training:

  Nexus 9300 Modules:

  - N9K-M12PQ (C9396PX, C9396TX, C93128PX, C93128TX)

  Nexus 9500 Modules:

- X9536PQ
- X9564PX
- X9564TX

- When you use a backslash (\) at end of a valid interface description, the parser identifies the backslash as a continuation character and appends an extra line break in command output by adding a new line character '\n' to the command string. This is a Day-1 behavior.
- Beginning with Cisco NX-OS Release 10.2(3)F, the **link-flap error-disable count** command can be configured on all physical ports on all Cisco Nexus 9000 Series switches.

Support for QSA

- 1 GB with QSA is *not* supported on Retimer Ports. For information on, see Retimer ports.
- Beginning with Cisco NX-OS Release 9.2(2), 10 GB with QSA is supported on the following ports:
  - Cisco Nexus 9336C-FX2 switch: Ports 1-36
  - Cisco Nexus 9364C switch: Ports 49-64
  - Cisco Nexus 9788TC line card: Ports 49-52
- Beginning with Cisco NX-OS Release 9.2(2), 1 GB with QSA is supported on the following ports:
  - Cisco Nexus 9336C-FX2 switch: Ports 7-32
  - Cisco Nexus 9364C switch: Ports 65 and 66 only

Guidelines for ethernet port speed and duplex mode

- You usually configure Ethernet port speed and duplex mode parameters to auto to allow the system to negotiate the speed and duplex mode between ports. If you decide to configure the port speed and duplex modes manually for these ports, consider the following:
  - Before you configure the speed and duplex mode for an Ethernet or management interface, see the Default Settings section for the combinations of speeds and duplex modes that can be configured at the same time.
  - If you set the Ethernet port speed to auto, the device automatically sets the duplex mode to auto.
  - If you enter the **no speed** command, the device automatically sets both the speed and duplex parameters to auto (the **no speed** command produces the same results as the **speed auto** command).
  - If you configure an Ethernet port speed to a value other than auto (for example, 1G, 10G, or 40G), you must configure the connecting port to match. Do not configure the connecting port to negotiate the speed.
  - Beginning with Cisco NX-OS Release 9.3(6), Cisco Nexus N9K-C92348GC-X switches support 10M full-duplex mode on ports 1 through 48.

Note The device cannot automatically negotiate the Ethernet port speed and duplex mode if the connecting port is configured to a value other than auto.

Caution Changing the Ethernet port speed and duplex mode configuration might shut down and reenable the interface.

- On Cisco Nexus 9000 Series Switches, the `show interface` and `show interface capability` commands may display 100 Mbps as a supported speed for certain ports. However, this speed is only supported when using the SFP-10G-T-X transceiver. For ports using GLC-TE transceivers, the lowest supported speed is 1 Gbps.

Support for autonegotiation

- Autonegotiation is *not* supported on 400G and 200G Copper links on these Nexus switches. Configure respective speed on the peer side to bring the link up.

Nexus switch	Copper support (No autonegotiation)	Release
N9K-C9348D-GX2A	400G	10.2(3)F
N9K-C9348D-GX2A	200G	10.3(3)F
N9K-C9364D-GX2A	400G	10.2(3)F
N9K-C9364D-GX2A	200G	10.3(3)F
N9K-C9332D-GX2B	400G	NX-OS 10.2(1q)F
N9K-C9332D-GX2B	200G	10.3(3)F
N9K-C93600CD-GX	400G	9.3(5)
N9K-C93600CD-GX	200G	10.3(3)F
N9K-C9316D-GX	400G	9.3(5)
N9K-C9316D-GX	200G	10.3(3)F
N9K-X9400-8D	400G	10.3(3)F
N9K-X9400-8D	200G	10.3(3)F

- Beginning with Cisco Nexus NX-OS Release 10.1(2), autonegotiation is supported for Speed 40G and 100G on these switches:
  - N9K-C93600CD-GX
  - N9K-C9316D-GX

- N9K-C9364C-GX

- Autonegotiation is *not* supported when N9K-C93108TC-FX3P switch is connected to either of the following switches:
  - N9K-C9236C, N9K-C92300YC, N9K-C93180YC-EX, N9K-C93180YC-EXU, N9K-C9232C, N9K-C92300YC, and N9K-C93180YC-FX.
  - N3K-C3172TQ-XL, N3K-C3172TQ-10GT, N3K-C3172PQ-10GE, and N3K-C3132Q-40GE.
- Beginning with Cisco NX-OS Release 9.2(2), autonegotiation (40 G/100 G) is supported on the following ports:
  - Cisco Nexus 9336C-FX2 switch: Ports 1-6 and 33-36
  - Cisco Nexus 9364C switch: Ports 49-64
  - Cisco Nexus 93240YC-FX2 switch: Ports 51-54
  - Cisco Nexus 9788TC line card: Ports 49-52
- Autonegotiation is not supported on 25G breakout ports.
- If cable length is more than 5 meters, autonegotiation is not supported. This cable length limitation is applicable only to copper cables and not applicable to optical cables.
- To configure speed, duplex, and automatic flow control for an Ethernet interface, you can use the **negotiate auto** command. To disable automatic negotiation, use the **no negotiate auto** command.
- For BASE-T copper ports, autonegotiation is enabled even when fixed speed is configured.

Cisco Cisco Nexus 9808

- Beginning with Cisco NX-OS Release 10.3(1)F, Interface Consistency Checker support is provided on Cisco Nexus 9800 platform switches.
- Beginning with Cisco NX-OS Release 10.3(1)F, native (400G, 100G, 40G) and breakout (4x100G) ports support is provided on N9K-X9836DM-A line card of Cisco Nexus 9800 platform switches.
- Beginning with Cisco NX-OS Release 10.3(1)F, 10G Optics support using CVR-QSFP-SFP10G adapter is provided for N9K-X9836DM-A line card of Cisco Nexus 9800 platform switches.
- Beginning with Cisco NX-OS Release 10.3(1)F, Auto negotiation is not supported for 40G, 100G copper based links for N9K-X9836DM-A line card of Cisco Nexus 9800 platform switches.
- Cisco Nexus 9808 platform switches have the following limitations for Physical Interface statistics support:
  - Port-channel is not supported
  - Broadcast counters/statistics are not supported for interface counters.
  - Locally generated/injected packets will not be classified into unicast, multicast or broadcast. However, these will be accounted under total packets and bytes. For example: cdp packets.
  - For **show interface ethernet 1/1 counters detailed snmp** command, Cisco Nexus 9800 platform supports different frame size range as below:

```
This platform counter Range
===============
TX Frame octet Range
TX legal frames with 1519-2500 bytes.
TX legal frames with 2501-9000 bytes.
Nexus existing platform
================
TX Length=1519-2047
TX Length=2048-4095
TX Length=4096-8191
TX Length=8192-9215
TX Length>=9216
Similar frame size suppport exists for Rx direction also.

show interface ethernet 1/1 counters detailed snmp
Ethernet1/1
Rx Packets: 4004
Rx Unicast Packets: 4000
Rx Jumbo Packets: 4000
Rx Bytes: 7031737
Rx Packets from 65 to 127 bytes: 1
Rx Packets from 128 to 255 bytes: 1
Rx Packets from 512 to 1023 bytes: 1
Rx Packets from 1024 to 1518 bytes: 1
Rx Packets from 1519 to 2500 bytes: 4000 >>>> New range supported
Tx Packets: 17
Tx Bytes: 4948
Tx Packets from 0 to 64 bytes: 2
Tx Packets from 65 to 127 bytes: 3
Tx Packets from 128 to 255 bytes: 10
Tx Packets from 512 to 1023 bytes: 1
Tx Packets from 1024 to 1518 bytes: 1
Tx Packets from 1519 to 2500 bytes: 2 >>>>> New range
```

- In case of interface error counters, Align-Err, Runs, Giants, Input discards and Output Discards counters are not supported and will be shown as 0.

  For example:

```
show interface ethernet 1/1 counters errors

--------------------------------------------------------------------------------
Port Align-Err FCS-Err Xmit-Err Rcv-Err UnderSize OutDiscards
--------------------------------------------------------------------------------
Eth1/1 0 0 0 0 0 0

--------------------------------------------------------------------------------
Port Single-Col Multi-Col Late-Col Exces-Col Carri-Sen Runts
--------------------------------------------------------------------------------
Eth1/1 0 0 0 0 0 0

--------------------------------------------------------------------------------
Port Giants SQETest-Err Deferred-Tx IntMacTx-Er IntMacRx-Er Symbol-Err
--------------------------------------------------------------------------------
Eth1/1 0 -- 0 0 0 0

--------------------------------------------------------------------------------
Port InDiscards
--------------------------------------------------------------------------------
Eth1/1 0

--------------------------------------------------------------------------------
Port Stomped-CRC
--------------------------------------------------------------------------------
Eth1/1 0
```

- Beginning with Cisco NX-OS Release 10.3(1)F, statistics support for Physical Interface is provided on Cisco Nexus 9808 platform switches.
- Beginning with Cisco NX-OS Release 10.3(1)F, UDLD support is provided on Cisco Nexus 9808 platform switches.

Retimer Ports

A retimer is an integrated device along the data plane path between the forwarding engine and front-panel port to help improve signal integrity. Additionally, retimers may also be used to recover clock timing and provide additional port features, such as MACsec or SyncE capabilities.

Retimer ports may experience longer link-up times depending on the negotiated speed, optics/transceiver and cable used as well as specific characteristics of the connected link partner. In many cases the link up time will not be more than a few seconds. However, in some cases, the link up time on retimer ports may be higher.

Table 6: Supported Retimer Ports

Switch or Line cards	Retimer Ports
N9K-X9788TC-FX	49-52
N9K-C93240YC-FX2 N9K-C93240YC-FX2-Z	51-54
N9K-C9336C-FX2	1-6, 33-36
N9K-C9364C	49-64
N9K-X96136YC-R	49-52
N9K-X9736C-FX	29-36
N9K-C9332C	25-32
N9K-C93180YC-FX3	1-54
N9K-C93216TC-FX2 N9K-C93360YC-FX2	97-108
N9K-X9716D-GX	1-16
N9K-C9336C-FX2-E	1-8
N9K-C9332D-GX2B	25-32
N9K-C9348D-GX2A	1-48
N9K-C9364D-GX2A	1-32

N9K-X9836DM-A	1-36
N9K-X9400-22L	1-22
N9K-X9400-16W	1-16
N9K-X9400-8D	1-8

Default Settings

The following lists the default settings for the basic interface parameters.

Parameter	Default
Description	Blank
Beacon	Disabled
Bandwidth	Data rate of interface
Throughput delay	100 microseconds
Administrative status	Shutdown
MTU	1500 bytes
UDLD global	Globally disabled
UDLD per-port enable state for fiber-optic media	Enabled on all Ethernet fiber-optic LAN ports
UDLD per-port enable state for copper media	Disabled on all Ethernet 1G, 10G, or 40G LAN ports
UDLD message interval	Disabled
UDLD aggressive mode	Disabled
Error disable	Disabled
Error disable recovery	Disabled
Error disable recovery interval	300 seconds
Buffer-boost	Enabled **Note** Feature available on N9K-X9564TX and N9K-X9564PX line cards and Cisco Nexus 9300 series devices.

Configuring the Basic Interface Parameters

When you configure an interface, you must specify the interface before you can configure its parameters.

Specifying the Interfaces to Configure

Before you begin

Before you can configure the parameters for one or more interfaces of the same type, you must specify the type and the identities of the interfaces.

The following table shows the interface types and identities that you should use for specifying the Ethernet and management interfaces.

Table 7: Information Needed to Identify an Interface for Configurations

Interface Type	Identity
Ethernet	I/O module slot numbers and port numbers on the module
Management	0 (for port 0)

The interface range configuration mode allows you to configure multiple interfaces with the same configuration parameters. After you enter the interface range configuration mode, all command parameters you enter are attributed to all interfaces within that range until you exit out of the interface range configuration mode.

You enter a range of interfaces using dashes (-) and commas (,). Dashes separate contiguous interfaces and commas separate noncontiguous interfaces. When you enter noncontiguous interfaces, you must enter the media type for each interface.

This example shows how to configure a contiguous interface range:

```
switch(config)# interface ethernet 2/29-30
switch(config-if-range)#
```

This example shows how to configure a noncontiguous interface range:

```
switch(config)# interface ethernet 2/29, ethernet 2/33, ethernet 2/35
switch(config-if-range)#
```

You can specify subinterfaces in a range only when the subinterfaces are on the same port, for example, 2/29.1-2. But you cannot specify the subinterfaces in a range of ports, for example, you cannot enter 2/29.2-2/30.2. You can specify two of the subinterfaces discretely, for example, you can enter 2/29.2, 2/30.2.

This example shows how to configure a a breakout cable:

```
switch(config)# interface ethernet 1/2/1
switch(config-if-range)#
```

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface*

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *interface*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
	Example:
	`switch(config)# interface mgmt0`
	`switch(config-if)#`
		Specifies the interface that you are configuring. You can specify the interface type and identity. For an Ethernet port, use **ethernet** *slot/port*. For the management interface, use **mgmt0**.
		Examples:
		• The 1st example shows how to specify the slot 2, port 1 Ethernet interface.
		• The 2nd example shows how to specify the management interface.
		Note
		You do not need to add a space between the interface type and identity (port or slot/port number) For example, for the Ethernet slot 4, port 5 interface, you can specify either “ethernet 4/5” or “ethernet4/5.” The management interface is either “mgmt0” or “mgmt 0.”
		When you are in the interface configuration mode, the commands that you enter configure the interface that you specified for this mode.

Configuring the Description

You can provide textual interface descriptions for the Ethernet and management interfaces.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface*
3. **description** *text*
4. **show interface** *interface*
5. **exit**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *interface*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
	Example:
	`switch(config)# interface mgmt0`
	`switch(config-if)#`
		Specifies the interface that you are configuring. You can specify the interface type and identity. For an Ethernet port, use **ethernet** *slot/port*. For the management interface, use **mgmt0**.
		Examples:
		• The 1st example shows how to specify the slot 2, port 1 Ethernet interface.
		• The 2nd example shows how to specify the management interface.

Step 3	**description** *text*
	Example:
	`switch(config-if)# description Ethernet port 3 on module 1`
	`switch(config-if)#`
		Specifies the description for the interface.

Step 4	**show interface** *interface* **Example:** `switch(config)# show interface ethernet 2/1`	(Optional) Displays the interface status, which includes the description parameter.

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set the interface description to Ethernet port 24 on module 3:

```
switch# configure terminal
switch(config)# interface ethernet 3/24
```

```
switch(config-if)# description server1
switch(config-if)#
```

The output of the **show interface eth** command is enhanced as shown in the following example:

```
Switch# show version
Software
BIOS: version 06.26
NXOS: version 6.1(2)I2(1) [build 6.1(2)I2.1]
BIOS compile time: 01/15/2014
NXOS image file is: bootflash:///n9000-dk9.6.1.2.I2.1.bin
NXOS compile time: 2/25/2014 2:00:00 [02/25/2014 10:39:03]

switch# show interface ethernet 6/36
Ethernet6/36 is up
admin state is up, Dedicated Interface
Hardware: 40000 Ethernet, address: 0022.bdf6.bf91 (bia 0022.bdf8.2bf3)
Internet Address is 192.168.100.1/24
MTU 9216 bytes, BW 40000000 Kbit, DLY 10 usec
```

Configuring the Beacon Mode

You can enable the beacon mode for an Ethernet port to flash its LED to confirm its physical location.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. [**no**] **beacon**
4. **show interface ethernet** *slot/port*
5. **exit**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an interface to configure, and enters interface configuration mode.

Step 3	[**no**] **beacon** **Example:**	Enables the beacon mode or disables the beacon mode. The default mode is disabled.

	Command or Action	Purpose
	`switch(config)# beacon` `switch(config-if)#`	

Step 4	**show interface ethernet** *slot/port*
	Example:
	`switch(config)# show interface ethernet 2/1`
	`switch(config-if)#`
		(Optional) Displays the interface status, which includes the beacon mode state.

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enable the beacon mode for the Ethernet port 3/1:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# beacon
switch(config-if)#
```

This example shows how to disable the beacon mode for the Ethernet port 3/1:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# no beacon
switch(config-if)#
```

This example shows how to configure the dedicated mode for Ethernet port 4/17 in the group that includes ports 4/17, 4/19, 4/21, and 4/23:

```
switch# configure terminal
switch(config)# interface ethernet 4/17, ethernet 4/19, ethernet 4/21, ethernet 4/23
switch(config-if)# shutdown
switch(config-if)# interface ethernet 4/17
switch(config-if)# no shutdown
switch(config-if)#
```

Configuring the Error-Disabled State

You can view the reason that an interface moves to the error-disabled state and configure automatic recovery.

Enabling the Error-Disable Detection

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

SUMMARY STEPS

1. **configure terminal**
2. **errdisable detect cause** {**acl-exception** | **all** | **link-flap** | **loopback**}
3. **shutdown**
4. **no shutdown**
5. **link-flap error-disable count** <*number_of_link_flaps*> **interval** <*time_in_seconds*>
6. **show interface status err-disabled**
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**errdisable detect cause** {**acl-exception** | **all** | **link-flap** | **loopback**}
	Example:
	`switch(config)# errdisable detect cause all`
	`switch(config-if)#`
		Specifies a condition under which to place the interface in an error-disabled state. The default is enabled.

Step 3	**shutdown**
	Example:
	`switch(config-if)# shutdown`
	`switch(config)#`
		Brings the interface down administratively. To manually recover the interface from the error-disabled state, enter this command first.

Step 4	**no shutdown**
	Example:
	`switch(config-if)# no shutdown`
	`switch(config)#`
		Brings the interface up administratively and enables the interface to recover manually from the error-disabled state.

Step 5	**link-flap error-disable count** <*number_of_link_flaps*> **interval** <*time_in_seconds*> **Example:** `switch(config-if)# link-flap error-disable count 10 interval 30`	Specifies the link-flap error-disable count and interval that you are configuring. • **count** configures the number of flaps that the port can tolerate. The maximum number of flaps that can be tolerated is 30 and minimum is 2.

	Command or Action	Purpose
		• **interval** configures the time in seconds for the configured flap count to occur after which action is taken. The maximum interval is 420 seconds and minimum is 30 seconds.
Step 6	**show interface status err-disabled** **Example:** `switch(config)# show interface status err-disabled`	(Optional) Displays information about error-disabled interfaces.
Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

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

```
switch(config)# errdisable detect cause all
switch(config)#
```

Enabling the Error-Disabled Recovery

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

SUMMARY STEPS

1. **configure terminal**
2. **errdisable recovery cause** {**all** | **bpduguard** | **failed-port-state** | **link-flap** | **loopback** | **miscabling** | **psecure-violation** | **security-violation** | **storm-control** | **udld** | **vpc-peerlink**}
3. **show interface status err-disabled**
4. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

	Command or Action	Purpose

Step 2	**errdisable recovery cause** {**all** | **bpduguard** | **failed-port-state** | **link-flap** | **loopback** | **miscabling** | **psecure-violation** | **security-violation** | **storm-control** | **udld** | **vpc-peerlink**}
	Example:
	`switch(config)# errdisable recovery cause all`
	`switch(config-if)#`
		Specifies a condition under which the interface automatically recovers from the error-disabled state, and the device retries bringing the interface up. The device waits 300 seconds to retry. The default is disabled.

Step 3	**show interface status err-disabled**
	Example:
	`switch(config)# show interface status err-disabled`
	`switch(config-if)#`
		(Optional) Displays information about error-disabled interfaces.

Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

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

```
switch(config)# errdisable recovery cause all
switch(config)#
```

Configuring the Error-Disabled Recovery Interval

You can configure the error-disabled recovery timer value.

SUMMARY STEPS

1. **configure terminal**
2. **errdisable recovery interval** *interval*
3. **show interface status err-disabled**
4. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

	Command or Action	Purpose

Step 2	**errdisable recovery interval** *interval*
	Example:
	`switch(config)# errdisable recovery interval 32`
	`switch(config-if)#`
		Specifies the interval for the interface to recover from the error-disabled state. The range is from 30 to 65535 seconds, and the default is 300 seconds.

Step 3	**show interface status err-disabled**
	Example:
	`switch(config)# show interface status err-disabled`
	`switch(config-if)#`
		(Optional) Displays information about error-disabled interfaces.

Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure the error-disabled recovery timer to set the interval for recovery to 32 seconds:

```
switch(config)# errdisable recovery interval 32
switch(config)#
```

Configuring the MDIX Parameter

To detect the type of connection (crossover or straight) with another copper Ethernet port, enable the medium dependent independent crossover (MDIX) parameter for the local port. By default, this parameter is enabled.

Before you begin

Enable MDIX for the remote port.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot* / *port*
3. {**mdix auto** | **no mdix**}
4. **show interface ethernet** *slot* / *port*
5. **exit**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot* / *port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an interface to configure, and enters into interface configuration mode.

Step 3	{**mdix auto** | **no mdix**}
	Example:
	`switch(config)# mdix auto`
	`switch(config-if)#`
	`switch(config)# no mdix`
	`switch(config-if)#`
		Specifies whether to enable or disable MDIX detection for the port.
		Note
		The **no mdix auto** command is supported only on N9K-C93108TC-EX, N9K-C93108TC-FX, N9K-X9788TC-FX, and N9K-C9348GC-FXP devices.

Step 4	**show interface ethernet** *slot* / *port*
	Example:
	`switch(config)# show interface ethernet 3/1`
	`switch(config-if)#`
		Displays the interface status, which includes the MDIX status.

Step 5	**exit** **Example:** `switch(config)# exit`	Exits the interface mode.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enable MDIX for Ethernet port 3/1:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# mdix auto
switch(config-if)#
```

This example shows how to enable MDIX for Ethernet port 3/1:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
```

```
switch(config-if)# no mdix
switch(config-if)#
```

Configuring Media-Type for SFP-10G-T-X

To specify the SFP-10G-T-X device connection on an interface, use the **media-type 10g-tx** command in interface configuration mode. To restore the default value, use the **no** form of this command.

SUMMARY STEPS

1. configure terminal
2. interface *interface-id*
3. media-type 10g-tx

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	configure terminal **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 2	interface *interface-id* **Example:** `Switch (config)# interface ethernet 1/5`	Specify the media port to be configured, and enter interface configuration mode.
Step 3	media-type 10g-tx **Example:** `Switch (Config)# [no] media-type 10g-tx`	Configure the SFP-10G-T-X device connection on an interface. **Note** Interface configured with media-type 10G-TX while in admin up state will remain errdisabled under Unsupported media-type. To remove this condition, use the following commands on the interface: • **shutdown** • **no shutdown**

Verifying Media-Type

This example shows how to verify the media-type configuration:

Note The ports that support SFP-10G-T-X may vary in different devices. This example displays the port numbers that support SFP-10G-T-X on a Cisco Nexus N9K-C93240YC-FX2 switch.

```
switch# sh running-config interface ethernet 1/2

!Command: show running-config interface Ethernet1/2
!Running configuration last done at: Mon Jun  1 10:16:46 2020
!Time: Mon Jun  1 10:16:54 2020

version 9.3(5) Bios:version 05.41

interface Ethernet1/2
  switchport
  switchport access vlan 10
  mtu 9216
  media-type 10g-tx
  no shutdown

Supported ports in Switch 01:
switch# sh interface status | i i SFP-10
Eth1/2        --                 connected 10        full    10G     SFP-10G-T-X
Eth1/6        --                 connected 11        full    10G     SFP-10G-T-X
Eth1/8        --                 connected 11        full    10G     SFP-10G-T-X
Eth1/12       --                 connected 12        full    10G     SFP-10G-T-X
Eth1/14       --                 connected 12        full    10G     SFP-10G-T-X
Eth1/18       --                 connected 13        full    10G     SFP-10G-T-X
Eth1/20       --                 connected 13        full    10G     SFP-10G-T-X
Eth1/24       --                 connected 14        full    10G     SFP-10G-T-X
Eth1/26       --                 connected 14        full    10G     SFP-10G-T-X
Eth1/30       --                 connected 15        full    10G     SFP-10G-T-X
Eth1/32       --                 connected 15        full    10G     SFP-10G-T-X
Eth1/36       --                 connected 16        full    10G     SFP-10G-T-X
Eth1/38       --                 connected 16        full    10G     SFP-10G-T-X
Eth1/42       --                 connected 20        full    10G     SFP-10G-T-X
Eth1/44       Connect_to_Sw_01   connected 202       full    10G     SFP-10G-T-X
Eth1/48       Connect_to_Sw_02   connected 202       full    10G     SFP-10G-T-X



switch# sh mod
Mod Ports             Module-Type                        Model           Status
--- ----- ------------------------------------ --------------------- ---------
1    60   48x10/25G + 12x40/100G Ethernet Modul N9K-C93240YC-FX2      active *

Mod  Sw                      Hw    Slot
---  ----------------------- ------ ----
1    9.3(4.104)              0.3020 NA


Mod  MAC-Address(es)                         Serial-Num
---  --------------------------------------  ----------
1    b4-de-31-94-4e-c8 to b4-de-31-94-4f-0f  FDO2143306S

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

Configuring the MTU Size

MTU is configured per interface, where the interface can be a Layer 2 or a Layer 3 interface. Every interface has default MTU of 1500 bytes. This value is called system default MTU. You can configure a Layer 2 interface, with a a value of 9216 bytes, which is the default value of the system jumbo MTU. To allow an

MTU value that is between 1500 and 9216, system jumbo MTU needs to be adjusted to appropriate value where interface can be configured with the same value.

Note You can change the system jumbo MTU size. When the value is changed, the Layer 2 interfaces that use the system jumbo MTU value, will automatically changes to the new system jumbo MTU value.

A Layer 3 interface, can be Layer 3 physical interface switch virtual interface (SVI), and subinterface, you can configure an MTU size between 576–9216 bytes.

Configuring the Interface MTU Size

For Layer 3 interfaces, you can configure an MTU with keyword MTU and value in bytes where value is between 576–9216 bytes. Beginning with Cisco NX-OS Release 9.3(1), you can configure the MTU size up to 9216 bytes on the management interfaces on all Cisco Nexus 9000 switches. The change in the configuration may trigger a temporary link flap at the end device.

For Layer 2 interfaces, you can configure an interface using the keyword MTU with value in bytes. The value can be a system default MTU size of 1500 bytes or the system jumbo MTU value that can be adjusted to the default size of 9216 bytes.

If you need to use a different system jumbo MTU size for Layer 2 interfaces, see the *Configuring the System Jumbo MTU Size* section.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*, vlan *vlan-id* mgmt *0*
3. **mtu** *size*
4. **exit**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port*, vlan *vlan-id* mgmt *0*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
	`switch(config)# interface vlan 100`
	`switch(config-if)#`
	`switch(config)# interface mgmt 0`
	`switch(config-if)#`
		Specifies an Ethernet interface to configure, and enters interface configuration mode.

	Command or Action	Purpose

Step 3	**mtu** *size*
	Example:
	`switch(config-if)# mtu 9216`
	`switch(config-if)#`
		Configure the MTU value on an interface.
		For a Layer 3 interface, a physical Layer 3 interface, an SVI or sub-interface, then the value can be between 576-9216 bytes. If the interface is a physical Layer 2 interface, then the value can be 1500 or system jumbo MTU value.

Step 4	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Example

This example shows how to configure the Layer 2 Ethernet port 3/1 with the default MTU size (1500):

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# switchport
switch(config-if)# mtu 1500
switch(config-if)#
```

This example displays the output of show running-config interface command:

```
switch# show run int mgmt0
!Command: show running-config interface mgmt0
!Running configuration last done at: Fri May 31 11:32:28 2019
!Time: Fri May 31 11:32:33 2019
version 9.3(1) Bios:version 07.65
interface mgmt0
mtu 9216
vrf member management
ip address 168.51.170.73/82
```

Configuring the System Jumbo MTU Size

You can configure and use the system jumbo MTU for a Layer 2 interfaces MTU value. The system jumbo MTU must be specified as an even number between 1500 and 9216. The default value of system jumbo MTU is 9216 bytes.

SUMMARY STEPS

1. **configure terminal**
2. **system jumbomtu** *size*
3. **interface** *type slot/port*
4. **mtu** *size*
5. **exit**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**system jumbomtu** *size*
	Example:
	`switch(config)# system jumbomtu 8000`
	`switch(config)#`
		Specifies the system jumbo MTU size. Use an even number between 1500 and 9216.

Step 3	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Specifies an interface to configure and enters interface configuration mode.

Step 4	**mtu** *size*
	Example:
	`switch(config-if)# mtu 8000`
	`switch(config-if)#`
		System jumbo MTU is added to a Layer 2 interface.

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure the system jumbo MTU as 8000 bytes and how to change the MTU specification for a Layer 2 interface that was configured with the previous jumbo MTU size:

```
switch# configure terminal
switch(config)# system jumbomtu 8000
switch(config)# interface ethernet 2/2
switch(config-if)# mtu 8000
```

Configuring the Bandwidth

You can configure the bandwidth for Ethernet interfaces. The physical layer uses an unchangeable bandwidth of 1G, 10G, or 40G, but you can configure a value of 1 to 100,000,000 KB for Level 3 protocols.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. **bandwidth** *kbps*
4. **show interface ethernet** *slot/port*
5. **exit**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an Ethernet interface to configure, and enters interface configuration mode.

Step 3	**bandwidth** *kbps*
	Example:
	`switch(config-if)# bandwidth 1000000`
	`switch(config-if)#`
		Specifies the bandwidth as an informational-only value between 1 and 100,000,000.

Step 4	**show interface ethernet** *slot/port* **Example:** `switch(config)# show interface ethernet 2/1`	(Optional) Displays the interface status, which includes the bandwidth value.

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure an informational value of 1,000,000 Kb for the Ethernet slot 3, port 1 interface bandwidth parameter:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# bandwidth 1000000
switch(config-if)#
```

Configuring the Throughput Delay

You can configure the interface throughput delay for Ethernet interfaces. The actual delay time does not change, but you can set an informational value between 1 and 16777215, where the value represents the number of tens of microseconds.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. **delay** *value*
4. **show interface ethernet** *slot/port*
5. **exit**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an Ethernet interface to configure, and enters interface configuration mode.

Step 3	**delay** *value*
	Example:
	`switch(config-if)# delay 10000`
	`switch(config-if)#`
		Specifies the delay time in tens of microseconds. You can set an informational value range between 1 and 16777215 tens of microseconds.

Step 4	**show interface ethernet** *slot/port* **Example:**	(Optional) Displays the interface status, which includes the throughput-delay time.

	Command or Action	Purpose
	`switch(config)# show interface ethernet 3/1` `switch(config-if)#`	

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure the throughput-delay time so that one interface is preferred over another. A lower delay value is preferred over a higher value. In this example, Ethernet 7/48 is preferred over 7/47. The default delay for 7/48 is less than the configured value on 7/47, which is set for the highest value (16777215):

```
switch# configure terminal
switch(config)# interface ethernet 7/47
switch(config-if)# delay 16777215
switch(config-if)# ip address 192.168.10.1/24
switch(config-if)# ip router eigrp 10
switch(config-if)# no shutdown
switch(config-if)# exit
switch(config)# interface ethernet 7/48
switch(config-if)# ip address 192.168.11.1/24
switch(config-if)# ip router eigrp 10
switch(config-if)# no shutdown
switch(config-if)#
```

Note You must first ensure the EIGRP feature is enabled by running the **feature eigrp** command.

Shutting Down and Activating the Interface

You can shut down and restart Ethernet or management interfaces. When you shut down interfaces, they become disabled and all monitoring displays show them as being down. This information is communicated to other network servers through all dynamic routing protocols. When the interfaces are shut down, the interface is not included in any routing updates. To activate the interface, you must restart the device.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface*
3. **shutdown**

4. **show interface** *interface*
5. **no shutdown**
6. **show interface** *interface*
7. **exit**
8. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *interface*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
	
	`switch(config)# interface mgmt0`
	`switch(config-if)#`
		Specifies the interface that you are configuring. You can specify the interface type and identity. For an Ethernet port, use *ethernet slot/port*. For the management interface, use *mgmt0*.
		Examples:
		• The 1st example shows how to specify the slot 2, port 1 Ethernet interface.
		• The 2nd example shows how to specify the management interface.

Step 3	**shutdown**
	Example:
	`switch(config-if)# shutdown`
	`switch(config-if)#`
		Disables the interface.

Step 4	**show interface** *interface*
	Example:
	`switch(config-if)# show interface ethernet 2/1`
	`switch(config-if)#`
		(Optional) Displays the interface status, which includes the administrative status.

Step 5	**no shutdown**
	Example:
	`switch(config-if)# no shutdown`
	`switch(config-if)#`
		Reenables the interface.

Step 6	**show interface** *interface*
	Example:
	`switch(config-if)# show interface ethernet 2/1`
	`switch(config-if)#`
		(Optional) Displays the interface status, which includes the administrative status.

	Command or Action	Purpose

Step 7	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 8	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to change the administrative status for Ethernet port 3/1 from disabled to enabled:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# shutdown
switch(config-if)# no shutdown
switch(config-if)#
```

Configuring the UDLD Mode

You can configure normal unidirectional link detection (UDLD) modes for Ethernet interfaces on devices configured to run UDLD.

Before you can enable the aggressive UDLD mode for an interface, you must make sure that UDLD is already enabled globally on the device and on the specified interfaces.

Note If the interface is a copper port, you must use the command enable UDLD to enable the UDLD. If the interface is a fiber port you need not explicitly enable UDLD on the interface. However if you attempt to enable UDLD on a fiber port using the enable UDLD command, you may get an error message indicating that is not a valid command.

The following table lists CLI details to enable and disable UDLD on different interfaces

Table 8: CLI Details to Enable or Disable UDLD on Different Interfaces

Description	Fiber port	Copper or Nonfiber port
Default setting	Enabled	Disabled
Enable UDLD command	**no udld disable**	**udld enable**
Disable UDLD command	**udld disable**	**no udld enable**

Before you begin

You must enable UDLD for the other linked port and its device.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **feature udld**
3. **udld message-time** *seconds*
4. **udld aggressive**
5. **interface ethernet** *slot/port*
6. **udld** [**enable** | **disable**]
7. **show udld** [**ethernet** *slot/port* | **global** | **neighbors**]
8. **exit**
9. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	[**no**] **feature udld**
	Example:
	`switch(config)# feature udld`
	`switch(config)#`
	`switch(config)# no feature udld`
	`switch(config)#`
		Enables/Disables UDLD for the device.

Step 3	**udld message-time** *seconds*
	Example:
	`switch(config)# udld message-time 30`
	`switch(config)#`
		(Optional) Specifies the interval between sending UDLD messages. The range is from 7 to 90 seconds, and the default is 15 seconds.

Step 4	**udld aggressive**
	Example:
	`switch(config)# udld aggressive`
	`switch(config)#`
		Enables UDLD in aggressive mode by default on all fiber interfaces. Use the **no** form to disable aggressive mode UDLD on all fibers ports by default.
		Note
		Use the udld aggressive command to configure the ports to use a UDLD mode:
		• To enable fiber interfaces for the aggressive mode, enter the **udld aggressive** command in the global command mode and all the fiber interfaces will be in aggressive UDLD mode.

	Command or Action	Purpose
		• To enable the copper interfaces for the aggressive mode, you must enter the **udld aggressive** command in the interface mode, specifying each interface you want in aggressive UDLD mode. To use the aggressive UDLD mode, you must configure the interfaces on both ends of the link for the aggressive UDLD mode.

Step 5	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		(Optional) Specifies an interface to configure, and enters interface configuration mode.

Step 6	**udld** [**enable** | **disable**]
	Example:
	`switch(config-if)# udld enable`
	`switch(config-if)#`
		Enables UDLD in normal mode by default on all fiber interfaces. Use the **no** form to disable normal mode UDLD on all fibers ports by default.

Step 7	**show udld** [**ethernet** *slot/port* | **global** | **neighbors**]
	Example:
	`switch(config)# show udld`
	`switch(config)#`
		(Optional) Displays the UDLD status.

Step 8	**exit**
	Example:
	`switch(config-if-range)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 9	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enable the UDLD for the device:

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

This example shows how to set the UDLD message interval to 30 seconds:

```
switch# configure terminal
switch(config)# feature udld
switch(config)# udld message-time 30
switch(config)#
```

This example shows how to disable UDLD for Ethernet port 3/1:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if-range)# no udld enable
switch(config-if-range)# exit
```

This example shows how to disable UDLD for the device:

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

This example shows how to enable fiber interfaces for the aggressive UDLD mode:

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

This example shows how to enable the aggressive UDLD mode for the copper Ethernet interface3/1:

```
switch# configure terminal
switch(config)# interface ethernet 3
switch(config-if)# udld aggressive
```

This example shows how to check if aggressive mode is enabled.

```
switch# sh udld global

UDLD global configuration mode: enabled-aggressive
UDLD global message interval: 15
switch#
```

This example shows how to check if udld aggressive mode is operational for a given interface.

```
switch# sh udld ethernet 8/2

Interface Ethernet8/2
------------------------------
Port enable administrative configuration setting: device-default
Port enable operational state: enabled-aggressive
Current bidirectional state: bidirectional
Current operational state:  advertisement - Single neighbor detected
Message interval: 15
Timeout interval: 5
<>
```

Configuring Debounce Timers

You can enable the debounce timer for Ethernet ports by specifying a debounce time (in milliseconds) or disable the timer by specifying a debounce time of 0.

Note The link state of 10G and 100G ports may change repeatedly when connected to service provider network. As a part of *link reset* or *break-link* functionality, it is expected that the Tx power light on the SFP to change to N/A state, at an event of link state change.

However, to prevent this behavior during the link state change, you may increase the link debounce timer to start from 500ms and increase it in 500ms intervals until the link stabilizes. On the DWDM, UVN, and WAN network, it is recommended to disable automatic link suspension (ALS) whenever possible. ALS suspends the link on the WAN when the Nexus turn off the link.

Note

The **link debounce time** and **link debounce link-up time** commands can only be applied to a physical Ethernet interface.

Use the **show interface debounce** command to display the debounce times for all Ethernet ports.

The **link debounce time** command is not supported on 10G and 40G ports on the Cisco Nexus 93300YC-FX and Cisco Nexus 9336C-FX switches.

The **link debounce time** command is supported on 1G, 10G, 40G, 25G and 100G SFP/QSFP ports on the Cisco Nexus 9000 series switches.

The **link debounce time** is supported on 1G, 10G, 25G, 40G and 100G ports on Cisco Nexus N9K-C9732C-FX, N9K-C9364C, N9K-X97160YC-EX, N9K-C9336C-FX2, and N9K-C93240YC-FX2 platform switches.

The **link debounce time** command is supported on 1G, 10G, 40G, 25G and 100G SFP/QSFP ports on the Cisco Nexus 9000 series switches.

The **link debounce time** is supported on 1G, 10G, 25G, 40G and 100G ports on Cisco Nexus N9K-C9732C-FX, N9K-C9364C, N9K-X97160YC-EX, N9K-C9336C-FX2, and N9K-C93240YC-FX2 platform switches.

The **link debounce time** is not supported on RJ-45 ports on Cisco Nexus 9500 platform switches with N9K-X97160TC-FX line cards.

Beginning with Cisco NX-OS Release 10.2(3)F, the **link debounce time** command is supported on N9K-C93180YC-FX3S, N9K-C93180YC-FX3, N9K-C93108TC-FX3P and N9K-X9716D-GX platform switches.

Beginning with Cisco NX-OS Release 10.2(3)F, the **link debounce time** command is supported on this following ports and platform switches:

Ports	Switches
1G	Cisco Nexus N9K-C9364C, N9K-C93300YC-FX2, N9K-C93240YC-FX2, N9K-C93240YC-FX2-Z, N9K-X97160YC-EX, N9K-C9316D-GX, N9K-C93600CD-GX, N9K-C9364C-GX, N9K-C9232C, N9K-C93180YC-EX, N9K-C93180YC-EXU, N9K-C93180YC-EX-24, N9K-C93180YC-FX3S, N9K-C93180YC-FX3, N9K-C93108TC-FX3P, and N9K-X9716D-GX
10G	Cisco Nexus N9K-C9364C, N9K-C93300YC-FX2, N9K-C93240YC-FX2, N9K-C93240YC-FX2-Z, N9K-X97160YC-EX, N9K-C9316D-GX, N9K-C93600CD-GX, N9K-C9364C-GX, N9K-C9232C, N9K-C93180YC-EX, N9K-C93180YC-EXU, N9K-C93180YC-EX-24, N9K-C93180YC-FX3S, N9K-C93180YC-FX3, N9K-C93108TC-FX3P, and N9K-X9716D-GX
25G	Cisco Nexus N9K-C93300YC-FX2, N9K-C93240YC-FX2, N9K-C93240YC-FX2-Z, N9K-X97160YC-EX, N9K-C9316D-GX, N9K-C93600CD-GX, N9K-C9364C-GX, N9K-C9232C, N9K-C93180YC-EX, N9K-C93180YC-EXU, N9K-C93180YC-EX-24, N9K-C93180YC-FX3S, N9K-C93180YC-FX3, N9K-C93108TC-FX3P, and N9K-X9716D-GX
40G 100G 400G	Cisco Nexus N9K-C9364C, N9K-X9732C-FX, N9K-C9336C-FX2, N9K-C93300YC-FX2, N9K-C93240YC-FX2, N9K-C93240YC-FX2-Z, N9K-X97160YC-EX, N9K-C9316D-GX, N9K-C93600CD-GX, N9K-C9364C-GX, N9K-C9232C, N9K-C93180YC-EX, N9K-C93180YC-EXU, N9K-C93180YC-EX-24, N9K-C93180YC-FX3S, N9K-C93180YC-FX3, N9K-C93108TC-FX3P, and N9K-X9716D-GX

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. **link debounce time** *time*
4. **link debounce link-up** *time*

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an Ethernet interface to configure, and enters interface configuration mode.

Step 3	**link debounce time** *time*
	Example:
	`switch(config-if)# link debounce time 1000`
	`switch(config-if)#`
		Enables the debounce timer for the specified time (1 to 5000 milliseconds).
		If you specify 0 milliseconds, the debounce timer is disabled.

Step 4	**link debounce link-up** *time*
	Example:
	`switch(config-if)# link debounce link-up 1000`
	`switch(config-if)#`
		Enables the link-up timer for the specified time (1000 to 10000 milliseconds). This command applies only if the port speeds are 10G, 25G, 40G and 100G.
		The default value of the timer is 0. If the value is set to 0 the interface will be up without any delay.
		Note
		The **no link debounce link-up** command also resets the value to 0.
		Note
		This command is supported only on Cisco Nexus N9K-X9732C-FX , N9K-C93300YC-FX, N9K-C9336C-FX2, N9K-C9364C and N9K-X97160YC-EX switches.

Example

- The following example enables the debounce timer and sets the debounce time to 1000 milliseconds for an Ethernet interface:

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

- The following example disables the debounce timer for an Ethernet interface:

```
switch# configure terminal
switch(config)# interface ethernet 1/4
switch(config-if)# link debounce time 0
```

- The following example sets the debounce link-up timer to 1000 milliseconds for an Ethernet interface:

```
switch# configure terminal
switch(config)# interface ethernet 1/4
switch(config-if)# link debounce link-up time 1000
```

Configuring Port Profiles

You can apply several configuration parameters to a range of interfaces simultaneously. All the interfaces in the range must be the same type. You can also inherit the configurations from one port profile into another port profile. The system supports four levels of inheritance.

Creating a Port Profile

You can create a port profile on the device. Each port profile must have a unique name across types and the network.

Note Port profile names can include only the following characters:

- a-z
- A-Z
- 0-9
- No special characters are allowed, except for the following:
  - .
  - -
  - _

SUMMARY STEPS

1. **configure terminal**
2. **port-profile** [**type** {**ethernet** | **interface-vlan** | **port-channel**}] *name*
3. **exit**
4. (Optional) **show port-profile**

5. (Optional) **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**port-profile** [**type** {**ethernet** | **interface-vlan** | **port-channel**}] *name*	Creates and names a port profile for the specified type of interface and enters the port-profile configuration mode.
Step 3	**exit**	Exits the port-profile configuration mode.
Step 4	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 5	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to create a port profile named test for ethernet interfaces:

```
switch# configure terminal
switch(config)# port-profile type ethernet test
switch(config-ppm)#
```

Entering Port-Profile Configuration Mode and Modifying a Port Profile

You can enter the port-profile configuration mode and modify a port profile. To modify the port profile, you must be in the port-profile configuration mode.

SUMMARY STEPS

1. **configure terminal**
2. **port-profile** [**type** {**ethernet** | **interface-vlan** | **port-channel**}] *name*
3. **exit**
4. (Optional) **show port-profile**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.

	Command or Action	Purpose
Step 2	**port-profile** [**type** {**ethernet** | **interface-vlan** | **port-channel**}] *name*	Enters the port-profile configuration mode for the specified port profile and allows you to add or remove configurations to the profile.
Step 3	**exit**	Exits the port-profile configuration mode.
Step 4	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 5	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to enter the port-profile configuration mode for the specified port profile and bring all the interfaces administratively up:

```
switch# configure terminal
switch(config)# port-profile type ethernet test
switch(config-ppm)# no shutdown
switch(config-ppm)#
```

Assigning a Port Profile to a Range of Interfaces

You can assign a port profile to an interface or to a range of interfaces. All the interfaces must be the same type.

SUMMARY STEPS

1. **configure terminal**
2. **interface** [**ethernet** *slot/port* | **interface-vlan** *vlan-id* | **port-channel** *number*]
3. **inherit port-profile** *name*
4. **exit**
5. (Optional) **show port-profile**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**interface** [**ethernet** *slot/port* | **interface-vlan** *vlan-id* | **port-channel** *number*]	Selects the range of interfaces.
Step 3	**inherit port-profile** *name*	Assigns the specified port profile to the selected interfaces.
Step 4	**exit**	Exits the port-profile configuration mode.

	Command or Action	Purpose
Step 5	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 6	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to assign the port profile named adam to Ethernet interfaces 7/3 to 7/5, 10/2, and 11/20 to 11/25:

```
switch# configure terminal
switch(config)# interface ethernet7/3-5, ethernet10/2, ethernet11/20-25
switch(config-if)# inherit port-profile adam
switch(config-if)#
```

Enabling a Specific Port Profile

To apply the port-profile configurations to the interfaces, you must enable the specific port profile. You can configure and inherit a port profile onto a range of interfaces before you enable that port profile. You would then enable that port profile for the configurations to take effect on the specified interfaces.

If you inherit one or more port profiles onto an original port profile, only the last inherited port profile must be enabled; the system assumes that the underlying port profiles are enabled.

You must be in the port-profile configuration mode to enable or disable port profiles.

SUMMARY STEPS

1. **configure terminal**
2. **port-profile** [**type** {**ethernet** | **interface-vlan** | **port-channel**}] *name*
3. **state enabled**
4. **exit**
5. (Optional) **show port-profile**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**port-profile** [**type** {**ethernet**	**interface-vlan**	**port-channel**}] *name*	Creates and names a port profile for the specified type of interface and enters the port-profile configuration mode.
Step 3	**state enabled**	Enables that port profile.
Step 4	**exit**	Exits the port-profile configuration mode.

	Command or Action	Purpose
Step 5	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 6	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to enter the port-profile configuration mode and enable the port profile:

```
switch# configure terminal
switch(config)# port-profile type ethernet test
switch(config-ppm)# state enabled
switch(config-ppm)#
```

Inheriting a Port Profile

You can inherit a port profile onto an existing port profile. The system supports four levels of inheritance.

SUMMARY STEPS

1. **configure terminal**
2. **port-profile** *name*
3. **inherit port-profile** *name*
4. **exit**
5. (Optional) **show port-profile**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**port-profile** *name*	Enters the port-profile configuration mode for the specified port profile.
Step 3	**inherit port-profile** *name*	Inherits another port profile onto the existing one. The original port profile assumes all the configurations of the inherited port profile.
Step 4	**exit**	Exits the port-profile configuration mode.
Step 5	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 6	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to inherit the port profile named adam onto the port profile named test:

```
switch# configure terminal
switch(config)# port-profile test
switch(config-ppm)# inherit port-profile adam
switch(config-ppm)#
```

Removing a Port Profile from a Range of Interfaces

You can remove a port profile from some or all of the interfaces to which you have applied the profile. You do this configuration in the interfaces configuration mode.

SUMMARY STEPS

1. **configure terminal**
2. **interface** [**ethernet** *slot/port* | **interface-vlan** *vlan-id* | **port-channel** *number*]
3. **no inherit port-profile** *name*
4. **exit**
5. (Optional) **show port-profile**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**interface** [**ethernet** *slot/port* | **interface-vlan** *vlan-id* | **port-channel** *number*]	Selects the range of interfaces.
Step 3	**no inherit port-profile** *name*	Un-assigns the specified port profile to the selected interfaces.
Step 4	**exit**	Exits the port-profile configuration mode.
Step 5	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 6	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to unassign the port profile named adam to Ethernet interfaces 7/3 to 7/5, 10/2, and 11/20 to 11/25:

```
switch# configure terminal
```

```
switch(config)# interface ethernet 7/3-5, 10/2, 11/20-25
switch(config-if)# no inherit port-profile adam
switch(config-if)#
```

Removing an Inherited Port Profile

You can remove an inherited port profile. You do this configuration in the port-profile mode.

SUMMARY STEPS

1. **configure terminal**
2. **port-profile** *name*
3. **no inherit port-profile** *name*
4. **exit**
5. (Optional) **show port-profile**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**port-profile** *name*	Enters the port-profile configuration mode for the specified port profile.
Step 3	**no inherit port-profile** *name*	Removes an inherited port profile from this port profile.
Step 4	**exit**	Exits the port-profile configuration mode.
Step 5	(Optional) **show port-profile**	Displays the port-profile configuration.
Step 6	(Optional) **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to remove the inherited port profile named adam from the port profile named test:

```
switch# configure terminal
switch(config)# port-profile test
switch(config-ppm)# no inherit port-profile adam
switch(config-ppm)#
```

Configuring link mac-up timer

This procedure describes how to configure mac up timers on DWDM/Dark fiber circuits.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **link mac-up timer** *seconds*

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet1/2`
	`switch(config-if)#`
		Configures an interface and enters interface configuration mode.

Step 3	**link mac-up timer** *seconds* **Example:** `switch(config-if)# link mac-up timer 10`	Enables modification of the link mac-up timer. The link mac-up timer range is 0-120. **Note** This should only be done on DWDM links.

Configuring 25G Autonegotiation

Autonegotiation allows devices to advertise enhanced modes of operation it possesses via the link segment and to detect corresponding enhanced operational modes that the other devices may be advertising. Autonegotiation provides the means to exchange information between two devices that share a link segment and to automatically configure both devices to take maximum advantage of their abilities.

Guidelines and Limitations for 25G Autonegotiation

- Beginning with Cisco NX-OS Release 9.2(1), autonegotiation on native 25G ports with copper cables is supported on Cisco Nexus N9K-X97160YC-EX, N9K-C93180YC-FX, N9K-C93240YC-FX2 and N9K-C93240YC-FX2-Z switches.
-
- Autonegotiation of 25G interfaces is disabled by default
- Copper-based 25G transceivers require autonegotiation. Enable the **command negotiate auto 25000** under a copper 25G interface. The interface may remain down if this parameter is mismatched between each end of the link.
- Autonegotiation is not supported on 25G breakout ports.

FEC selection with 25G Autonegotiation

Table 9: FEC Selection with 25G Autonegotiation

Hardware	FEC based on CR Lengths			
	1m	2m	3m	5m
N9K-C93240YC-FX2	No FEC	No FEC	FC-FEC	RS-IEEE
N9K-C93180YC-FX	No FEC	No FEC	FC-FEC	RS-IEEE
N9K-C93180YC-EX	No FEC	No FEC	FC-FEC	FC-FEC
N9K-X97160YC-EX	No FEC	No FEC	FC-FEC	FC-FEC

Enabling Autonegotiation

You can enable autonegotiation using the *negotiate auto* command. To enable autonegotiation, follow these steps:

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *port number*
3. **negotiate auto** *port speed*

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *port number*
	Example:
	`switch# int e1/7`
	`switch(config-if)#`
		Selects the interface and enters interface mode.

Step 3	**negotiate auto** *port speed*
	Example:
	`switch(config-if)# negotiate auto 25000`
	`switch(config-if)#`
		Enables autonegotiation on the selected interface.
		Note
		You must apply this command on interfaces at both sides of the 25G native link.

This example shows how to enable autonegotiation on a specified interface:

Example

```
switch# sh int e1/7 st
--------------------------------------------------------------------------------
Port          Name               Status    Vlan      Duplex  Speed   Type
--------------------------------------------------------------------------------
Eth1/7        --                 connected routed    full    25G     SFP-H25GB-CU1M
switch# conf
switch(config)# int e1/7
switch(config-if)# negotiate auto 25000
```

Disabling Autonegotiation

You can disable autonegotiation using the *no negotiate auto* command. To disable autonegotiation, follow these steps:

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *port number*
3. **no negotiate auto** *port speed*

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *port number*
	Example:
	`switch# int e1/7`
	`switch(config-if)#`
		Selects the interface and enters interface mode.

Step 3	**no negotiate auto** *port speed*
	Example:
	`switch(config-if)# no negotiate auto 25000`
	`switch(config-if)#`
		Disables autonegotiation on the selected interface.
		Note
		You must apply this command on interfaces at both sides of the link.

This example shows how to disable autonegotiation on a specified interface.

Example

```
switch# sh int e1/7 st
--------------------------------------------------------------------------------
Port          Name               Status    Vlan      Duplex  Speed   Type
--------------------------------------------------------------------------------
Eth1/7        --                 connected routed    full    25G     SFP-H25GB-CU1M
switch# conf
switch(config)# int e1/7
switch(config-if)# no negotiate auto 25000
```

Verifying the Basic Interface Parameters

You can verify the basic interface parameters by displaying their values. You can also clear the counters listed when you display the parameter values.

To display basic interface configuration information, perform one of the following tasks:

Command	Purpose
show cdp all	Displays the CDP status.
show interface *interface*	Displays the configured states of one or all interfaces.
show interface brief	Displays a table of interface states.
show interface status err-disabled	Displays information about error-disabled interfaces.
show udld *interface*	Displays the UDLD status for the current interface or all interfaces.
show udld global	Displays the UDLD status for the current device.

Monitoring the Interface Counters

You can display and clear interface counters using Cisco NX-OS.

Displaying Interface Statistics

You can set up to three sampling intervals for statistics collections on interfaces.

SUMMARY STEPS

1. **configure terminal**
2. **interface ether** *slot/port*
3. **load-interval counters** [**1** | **2** | **3**] *seconds*
4. **show interface** *interface*
5. **exit**

6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ether** *slot/port*
	Example:
	`switch(config)# interface ether 4/1`
	`switch(config)#`
		Specifies interface.

Step 3	**load-interval counters** [**1** | **2** | **3**] *seconds*
	Example:
	`switch(config)# load-interval counters 1 100`
	`switch(config)#`
		Sets up to three sampling intervals to collect bit-rate and packet-rate statistics. The default values for each counter is as follows:
		1—30 seconds (60 seconds for VLAN)
		2—300 seconds
		3—not configured

Step 4	**show interface** *interface*
	Example:
	`switch(config)# show interface ethernet 2/2`
	`switch#`
		(Optional) Displays the interface status, which includes the counters.

Step 5	**exit**
	Example:
	`switch(config-if-range)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set the three sample intervals for the Ethernet port 3/1:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# load-interval counter 1 60
switch(config-if)# load-interval counter 2 135
```

```
switch(config-if)# load-interval counter 3 225
switch(config-if)#
```

Clearing Interface Counters

You can clear the Ethernet and management interface counters by using the **clear counters interface** command. You can perform this task from the configuration mode or interface configuration mode.

SUMMARY STEPS

1. **clear counters interface** [**all** | **ethernet** *slot/port* | **loopback** *number* | **mgmt** *number* | **port channel** *channel-number*]
2. **show interface** *interface*
3. **show interface** [**ethernet** *slot/port* | **port channel** *channel-number*] **counters**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**clear counters interface** [**all** | **ethernet** *slot/port* | **loopback** *number* | **mgmt** *number* | **port channel** *channel-number*]
	Example:
	`switch# clear counters ethernet 2/1`
	`switch#`
		Clears the interface counters.

Step 2	**show interface** *interface*
	Example:
	`switch# show interface ethernet 2/1`
	`switch#`
		(Optional) Displays the interface status.

Step 3	**show interface** [**ethernet** *slot/port* | **port channel** *channel-number*] **counters**
	Example:
	`switch# show interface ethernet 2/1 counters`
	`switch#`
		(Optional) Displays the interface counters.

Example

This example shows how to clear the counters on Ethernet port 5/5:

```
switch# clear counters interface ethernet 5/5
switch#
```

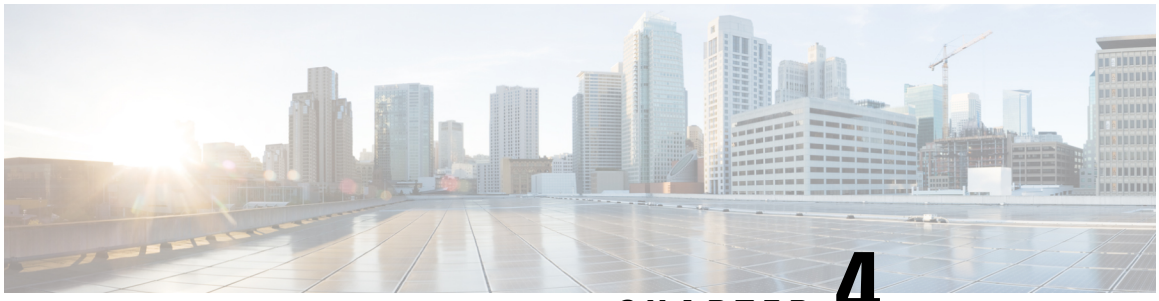
Configuration Example for QSA

For a Cisco Nexus 9396PX:

- Using the default configuration on port 2/1, all the QSFPs in port group 2/1-6 are brought up with a speed of 40G. If there are any QSA modules in port group 2/1-6, they are error disabled.
- Using the **speed-group** [**10000** | **40000**] command to configure port 2/7, all the QSAs in port group 2/7-12 are brought up with a speed of 10G or 40G. If there are any QSFP modules in port group 2/7-12, they are error disabled.

This example shows how to configure QSA for the first port in the speed group for a Cisco Nexus 9396PX:

```
switch# conf t
switch(config)# interface ethernet 2/7
switch(config-if)# speed-group 10000
```



CHAPTER 4

Configuring Layer 2 Interfaces

Information About Access and Trunk Interfaces

Note See the Cisco Nexus 9000 Series NX-OS High Availability and Redundancy Guide for complete information on high-availability features.

Note The device supports only IEEE 802.1Q-type VLAN trunk encapsulation.

About Access and Trunk Interfaces

A Layer 2 port can be configured as an access or a trunk port as follows:

- An access port can have only one VLAN configured on that port; it can carry traffic for only one VLAN.
- A trunk port can have two or more VLANs configured on that port; it can carry traffic for several VLANs simultaneously.

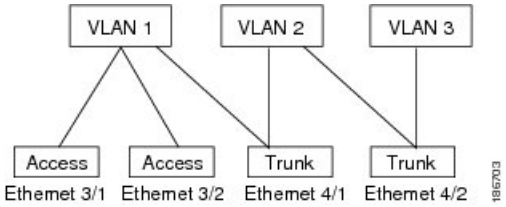
By default, all the ports on Cisco Nexus 9300-EX switches are Layer 3 ports and all the ports on Cisco Nexus 9300 switches are Layer 2 ports.

You can make all ports Layer 2 ports using the setup script or by entering the **system default switchport** command. See the Cisco Nexus 9000 Series NX-OS Fundamentals Configuration Guide for information about using the setup script. To configure the port as a Layer 2 port using the CLI, use the **switchport** command.

All ports in the same trunk must be in the same VDC, and trunk ports cannot carry VLANs from different VDCs.

The following figure shows how you can use trunk ports in the network. The trunk port carries traffic for two or more VLANs.

Figure 2: Trunk and Access Ports and VLAN Traffic



Note See the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide for information about VLANs.

In order to correctly deliver the traffic on a trunk port with several VLANs, the device uses the IEEE 802.1Q encapsulation, or tagging, method (see the “IEEE 802.1Q Encapsulation” section for more information).

Note See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for information about subinterfaces on Layer 3 interfaces.

To optimize the performance on access ports, you can configure the port as a host port. Once the port is configured as a host port, it is automatically set as an access port, and channel grouping is disabled. Use the host designation to decrease the time that it takes the designated port to begin to forward packets.

Only an end station can be set as a host port; you will receive an error message if you attempt to configure other ports as hosts.

If an access port receives a packet with an 802.1Q tag in the header other than the access VLAN value, that port drops the packet without learning its MAC source address.

A Layer 2 interface can function as either an access port or a trunk port; it cannot function as both port types simultaneously.

When you change a Layer 2 interface back to a Layer 3 interface, that interface loses all the Layer 2 configuration and resumes the default VLAN configurations.

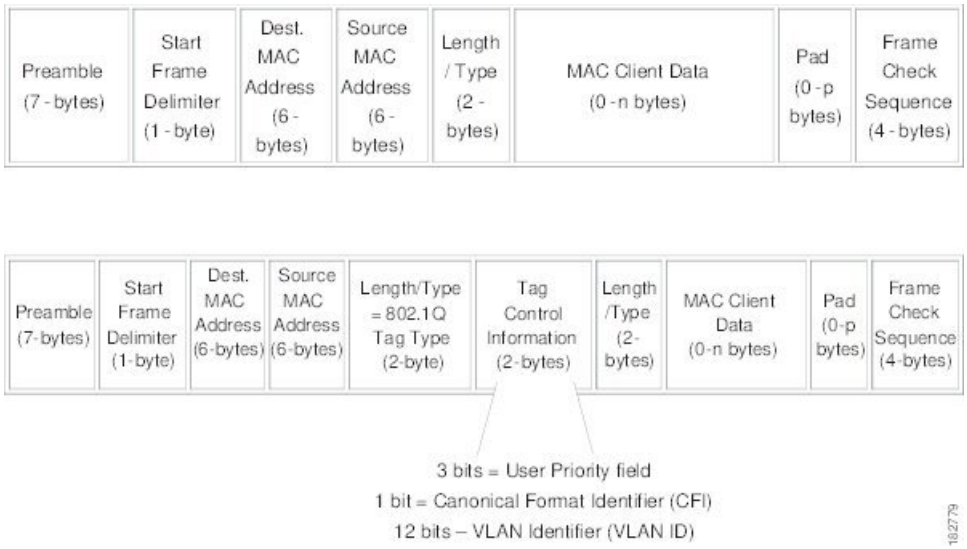
IEEE 802.1Q Encapsulation

Note For information about VLANs, see the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide.

A trunk is a point-to-point link between the switch and another networking device. Trunks carry the traffic of multiple VLANs over a single link and allow you to extend VLANs across an entire network.

To correctly deliver the traffic on a trunk port with several VLANs, the device uses the IEEE 802.1Q encapsulation, or tagging, method that uses a tag that is inserted into the frame header. This tag carries information about the specific VLAN to which the frame and packet belong. This method allows packets that are encapsulated for several different VLANs to traverse the same port and maintain traffic separation between the VLANs. Also, the encapsulated VLAN tag allows the trunk to move traffic end-to-end through the network on the same VLAN.

Figure 3: Header Without and With 802.1Q Tag



Drop Eligible Indicator

When Nexus 9000 switch receives a frame with DEI bit set to 1, it is forwarded as is to the next hop. For example, if the next hop is Nexus 6000, it drops frames on receiving a packet with the DEI bit set to 1 in the dot1q header.

Beginning with Cisco Nexus NX-OS release 10.2(3)F, the DEI bit is cleared whenever a frame is received with DEI bit set to 1.

The following is the configuration for resetting the DEI bit.

```
switch# conf t
Enter configuration commands, one per line. End with CNTL/Z.
switch(config)# system default reset-dei
switch(config)
```

The following is the configuration for setting the DEI bit.

```
switch# conf t
Enter configuration commands, one per line. End with CNTL/Z.
switch(config)# no system default reset-dei
switch(config)
```

Access VLANs

When you configure a port in access mode, you can specify which VLAN will carry the traffic for that interface. If you do not configure the VLAN for a port in access mode, or an access port, the interface carries traffic for the default VLAN (VLAN1).

You can change the access port membership in a VLAN by specifying the new VLAN. You must create the VLAN before you can assign it as an access VLAN for an access port. If you change the access VLAN on an access port to a VLAN that is not yet created, the system shuts that access port down.

If an access port receives a packet with an 802.1Q tag in the header other than the access VLAN value, that port drops the packet without learning its MAC source address.

Native VLAN IDs for Trunk Ports

A trunk port can carry nontagged packets simultaneously with the 802.1Q tagged packets. When you assign a default port VLAN ID to the trunk port, all untagged traffic travels on the default port VLAN ID for the trunk port, and all untagged traffic is assumed to belong to this VLAN. This VLAN is referred to as the native VLAN ID for a trunk port. That is, the native VLAN ID is the VLAN that carries untagged traffic on trunk ports.

Note Native VLAN ID numbers must match on both ends of the trunk.

The trunk port sends an egressing packet with a VLAN that is equal to the default port VLAN ID as untagged; all the other egressing packets are tagged by the trunk port. If you do not configure a native VLAN ID, the trunk port uses the default VLAN.

Note You cannot use a Fibre Channel over Ethernet (FCoE) VLAN as a native VLAN for an Ethernet trunk switchport.

Tagging Native VLAN Traffic

The Cisco software supports the IEEE 802.1Q standard on trunk ports. In order to pass untagged traffic through the trunk ports, you must create a VLAN that does not tag any packets (or you can use the default VLAN). Untagged packets can pass through trunk ports and access ports.

However, all packets that enter the device with an 802.1Q tag that matches the value of the native VLAN on the trunk are stripped of any tagging and egress the trunk port as untagged packets. This situation can cause problems because you may want to retain the tagging on packets on the native VLAN for the trunk port.

You can configure the device to drop all untagged packets on the trunk ports and to retain the tagging of packets entering the device with 802.1Q values that are equal to that of the native VLAN ID. All control traffic still passes on the native VLAN. This configuration is global; trunk ports on the device either do or do not retain the tagging for the native VLAN.

Allowed VLANs

By default, a trunk port sends traffic to and receives traffic from all VLANs. All VLAN IDs are allowed on each trunk. However, you can remove VLANs from this inclusive list to prevent traffic from the specified VLANs from passing over the trunk. Later, you can add any specific VLANs that you may want the trunk to carry traffic for back to the list.

To partition the Spanning Tree Protocol (STP) topology for the default VLAN, you can remove VLAN1 from the list of allowed VLANs. Otherwise, VLAN1, which is enabled on all ports by default, will have a very big STP topology, which can result in problems during STP convergence. When you remove VLAN1, all data traffic for VLAN1 on this port is blocked, but the control traffic continues to move on the port.

Note See the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide for more information about STP.

Note You can change the block of VLANs reserved for internal use. See the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide for more information about changing the reserved VLANs.

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, VLAN network, tunnel, and the port-channel interface.

Note A maximum of eight ports can be selected for the default interface. The default interfaces feature is not supported for management interfaces because the device could go to an unreachable state.

Switch Virtual Interface and Autostate Behavior

In Cisco NX-OS, a switch virtual interface (SVI) represents a logical interface between the bridging function and the routing function of a VLAN in the device.

The operational state of this interface is governed by the state of the various ports in its corresponding VLAN. An SVI interface on a VLAN comes up when at least one port in that VLAN is in the Spanning Tree Protocol (STP) forwarding state. Similarly, this interface goes down when the last STP forwarding port goes down or goes to another STP state.

High Availability

See the Cisco Nexus 9000 Series NX-OS High Availability and Redundancy Guide for complete information about high availability features.

Counter Values

See the following information on the configuration, packet size, incremented counter values, and traffic.

Configuration	Packet Size	Incremented Counters	Traffic
L2 port – without any MTU configuration	6400 and 10000	Jumbo, giant, and input error	Dropped
L2 port – with jumbo MTU 9216 in network-qos configuration	6400	Jumbo	Forwarded
L2 port – with jumbo MTU 9216 in network-qos configuration	10000	Jumbo, giant, and input error	Dropped
Layer 3 port with default Layer 3 MTU and jumbo MTU 9216 in network-qos configuration	6400	Jumbo	Packets are punted to the CPU (subjected to CoPP configs), get fragmented, and then they are forwarded by the software.
Layer 3 port with default Layer 3 MTU and jumbo MTU 9216 in network-qos configuration	6400	Jumbo	Packets are punted to the CPU (subjected to CoPP configs), get fragmented, and then they are forwarded by the software.
Layer 3 port with default Layer 3 MTU and jumbo MTU 9216 in network-qos configuration	10000	Jumbo, giant, and input error	Dropped
Layer 3 port with jumbo Layer 3 MTU and jumbo MTU 9216 in network-qos configuration	6400	Jumbo	Forwarded without any fragmentation.
Layer 3 port with jumbo Layer 3 MTU and jumbo MTU 9216 in network-qos configuration	10000	Jumbo, giant, and input error	Dropped
Layer 3 port with jumbo Layer 3 MTU and default L2 MTU configuration	6400 and 10000	Jumbo, giant, and input error	Dropped

Note

- Under 64 bytes packet with good CRC–The short frame counter increments.
- Under 64 bytes packet with bad CRC–The runts counter increments.
- Greater than 64 bytes packet with bad CRC–The CRC counter increments.

Prerequisites for Layer 2 Interfaces

Layer 2 interfaces have the following prerequisites:

- By default, Cisco NX-OS configures Layer 3 parameters. If you want to configure Layer 2 parameters, you need to switch the port mode to Layer 2. You can change the port mode by using the **switchport** command.
- You must configure the port as a Layer 2 port before you can use the **switchport mode** command. By default, all ports on the device are Layer 3 ports. By default, all ports on the Cisco Nexus 9504 and Cisco Nexus 9508 devices are Layer 2 ports.

Guidelines and Limitations for Layer 2 Interfaces

VLAN trunking has the following configuration guidelines and limitations:

- Cisco Nexus 9000 Series switches have the **vlan dot1q tag native** command that can be configured globally. This tags the native VLAN on the configured trunk ports. However, connected switches such as Catalyst 6500 or third-party switches, probably would not have a similar configuration enabled. This could result in unexpected behaviors. Therefore, it is recommended to have the **vlan dot1q tag native** command disabled in case the connected switch does not have it configured.
- BFD session on SVI interface with native VLAN is not supported with **vlan dot1q tag native** command configuration on Cisco Nexus 9300-X Cloud Scale Switches.
- Auto-negotiation is not supported on Cisco Nexus 9508 platform switches with N9K-X9636C-R, N9K-X9636C-RX, and N9K-X9636Q-R lines cards.
- Auto-negotiation is supported only on 10/25/40/100 direct attach copper cables.
- Auto-negotiation cannot be disabled on BaseT ports.
- Auto-negotiation is *not* supported on fiber based optics.
- Beginning with Cisco NX-OS Release 9.2(1), the Cisco Nexus 9508 platform switches with N9K-X96136YC-R line cards support 1 Gigabit speed on all 48 ports. However, because the auto negotiation is not supported, 1000BASE-T SFPs links comes up even the cable is removed.
- Beginning with Cisco NX-OS Release 9.2(1), auto negotiation on native 25G ports is supported on Cisco Nexus N9K-X97160YC-EX, N9K-C93180YC-FX, N9K-C93240YC-FX2 and N9K-C93240YC-FX2-Z switches.

Note Auto negotiation is not supported on Cisco Nexus N9K-C92300YC switch

- **show** commands with the **internal** keyword are not supported.
- Auto-negotiation is not supported on 25-G Ethernet transceiver modules on Cisco Nexus 9200 and 9300-FX platform switches, and Cisco Nexus 9500 platform switches that use N9K-X9700-EX line cards.
- On the Cisco Nexus 9364C switches, auto-negotiation might not work on ports 49-64 when bringing up 100G links using the QSFP-100G-CR4 cable. The workaround for this issue is that you must hard code the speed on ports 49-64 and disable auto-negotiation.
- Autonegotiation (40 G/100 G) and 1 GB with QSA is not supported on the following ports:
  - Cisco Nexus 9336C-FX2 switch: ports 1-6 and 33-36
  - Cisco Nexus 9364C switch: ports 49-66
  - Cisco Nexus 93240YC-FX2 switch: ports 51-54
  - Cisco Nexus 9788TC line card: ports 49-52

Note Peer speed must be set when using copper cables on these ports.

- On Cisco Nexus 9300 platform switches, a unicast ARP request to SVI is flooded to the other ports within the VLAN.
- ASE2 and ASE3 based Cisco Nexus 9000 Series switches acting as transit switches do not preserve the inner tag for double-tagged packets.

  The following CLI is mandatory only on LSE based Cisco Nexus 9000 Series switches. For seamless packet forwarding and preservation of all VLAN tags on pure transit boxes in the SP cloud that have no Q-in-Q encapsulation or decapsulation requirement, configure the CLI command, **system dot1q-tunnel transit** . To remove the CLI, use **no system dot1q-tunnel transit** CLI command.

  The caveats with the CLI that is executed on the switches are:
  - L2 frames that egress out of the trunk ports are tagged even on the native VLAN on the port.
  - Any other tunneling mechanism, for example, VXLAN and MPLS does not work with the CLI configured.
- A port can be either a Layer 2 or a Layer 3 interface; it cannot be both simultaneously.
- When you change a Layer 3 port to a Layer 2 port or a Layer 2 port to a Layer 3 port, all layer-dependent configuration is lost. When you change an access or trunk port to a Layer 3 port, all information about the access VLAN, native VLAN, allowed VLANs, and so forth, is lost.
- Do not connect devices with access links because access links may partition a VLAN.

- When connecting Cisco devices through an 802.1Q trunk, make sure that the native VLAN for an 802.1Q trunk is the same on both ends of the trunk link. If the native VLAN on one end of the trunk is different from the native VLAN on the other end, spanning tree loops might result.
- Disabling spanning tree on the native VLAN of an 802.1Q trunk without disabling spanning tree on every VLAN in the network can cause spanning tree loops. You must leave spanning tree enabled on the native VLAN of an 802.1Q trunk. If you cannot leave spanning tree enabled, you must disable spanning tree on every VLAN in the network. Make sure that your network has no physical loops before you disable spanning tree.
- When you connect two Cisco devices through 802.1Q trunks, the devices exchange spanning tree bridge protocol data units (BPDUs) on each VLAN allowed on the trunks. The BPDUs on the native VLAN of the trunk are sent untagged to the reserved IEEE 802.1D spanning tree multicast MAC address (01-80-C2-00-00-00). The BPDUs on all other VLANs on the trunk are sent tagged to the reserved Cisco Shared Spanning Tree (SSTP) multicast MAC address (01-00-0c-cc-cc-cd).
- Non-Cisco 802.1Q devices maintain only a single instance of spanning tree (the Mono Spanning Tree) that defines the spanning tree topology for all VLANs. When you connect a Cisco switch to a non-Cisco switch through an 802.1Q trunk, the Mono Spanning Tree of the non-Cisco switch and the native VLAN spanning tree of the Cisco switch combine to form a single spanning tree topology known as the Common Spanning Tree (CST).
- Because Cisco devices transmit BPDUs to the SSTP multicast MAC address on VLANs other than the native VLAN of the trunk, non-Cisco devices do not recognize these frames as BPDUs and flood them on all ports in the corresponding VLAN. Other Cisco devices connected to the non-Cisco 802.1Q cloud receive these flooded BPDUs. This BPDU reception allows Cisco switches to maintain a per-VLAN spanning tree topology across a cloud of non-Cisco 802.1Q devices. The non-Cisco 802.1Q cloud that separates the Cisco devices is treated as a single broadcast segment between all devices connected to the non-Cisco 802.1Q cloud through 802.1Q trunks.
- Make certain that the native VLAN is the same on all of the 802.1Q trunks that connect the Cisco devices to the non-Cisco 802.1Q cloud.
- If you are connecting multiple Cisco devices to a non-Cisco 802.1Q cloud, all of the connections must be through 802.1Q trunks. You cannot connect Cisco devices to a non-Cisco 802.1Q cloud through access ports because doing so places the access port on the Cisco device into the spanning tree “port inconsistent” state and no traffic will pass through the port.
- You can group trunk ports into port-channel groups, but all trunks in the group must have the same configuration. When a group is first created, all ports follow the parameters set for the first port to be added to the group. If you change the configuration of one of these parameters, the device propagates that setting to all ports in the group, such as the allowed VLANs and the trunk status. For example, if one port in a port group ceases to be a trunk, all ports cease to be trunks.
- If you try to enable 802.1X on a trunk port, an error message appears, and 802.1X is not enabled. If you try to change the mode of an 802.1X-enabled port to trunk, the port mode is not changed.
- Only ingress unicast packet counters are supported for SVI counters.
- When MAC addresses are cleared on a VLAN with the clear mac address-table dynamic command, the dynamic ARP (Address Resolution Protocol) entries on that VLAN are refreshed.
- If a static ARP entry exists on the VLAN and no MAC address to port mapping is present, the supervisor may generate an ARP request to learn the MAC address. Upon learning the MAC address, the adjacency entry points to the correct physical port.

- Cisco NX-OS does not support transparent bridging between two VLANs when one of the SVIs is on the Cisco Nexus 9000 using the BIA MAC (burned-in MAC address). This occurs when the BIA MAC is shared between SVIs/VLANs. A MAC, different from the BIA MAC, can be configured under the SVI for transparent bridging to work properly.

Note This behavior is applicable to Cisco Nexus 9300 Switches (Network Forwarding Engine) and Cisco Nexus 9500 Switches with 95xx,96xx,94xx line cards. This behavior is not applicable to Cisco Nexus 9200 Switches, Cisco Nexus 9300-EX and Cisco Nexus 9500 Switches with 9700-EX line cards.

- Port-local VLANs do not support Fabric Extenders (FEX).
- On Cisco Nexus 9364C switches, auto-negotiation may not work on ports 49-64 when bringing up 100G links using QSFP-100G-CR4 cable. To workaround this issue, you must hard-code the speed on ports 49-64 and disable auto-negotiation.
- You may get an error message when you attempt to configure the interface mode to trunk and trunk VLANs simultaneously. On Cisco NX-OS interfaces, the default value of interface mode is access. To implement any trunk related configurations, you must first change the interface mode to trunk and then configure the trunk VLAN ranges.
- On a vPC set up, if the VLAN is a vPC VLAN, the MAC address limit for VLAN and system is not supported.
- All the existing MACs may be flushed and relearnt, when the MAC address table limit is enabled for an interface, VLAN, and/or system.
- MAC address table limit enabled on vPC PO must be consistent across both the peers.
- If you configure MAC address table limit on system, port and VLAN at a time or in any combinations, each one of them will limit the MACs as they are configured. The preference will always be in the following order:
  - Port
  - VLAN
  - System
- MAC address table limit is not supported on vPC Peer-Links.
- Minimum configurable MAC address table limit is 100 and the maximum configurable limit is 196000.
- When an interface or a VLAN is removed from the set-up, the associated MAC address table limit configuration also gets removed.
- MAC address table limits are not supported on PVLAN interface types.
- When the MAC address table limit exceeds, it floods the traffic, by default.
- When you plug-in a FET-10G Fabric Extender Transceiver in a port on a Cisco Nexus N9K-C93180YC-FX3S switch or Cisco Nexus 9500 switch with N9K-X9716D-GX line card, you may see the links go up even if the ports are not converted to fabric ports using the command **switchport mode fex-fabric**.

- Beginning with Cisco NX-OS Release 10.2(1q)F, Layer 2 (L2) interfaces are supported on the N9K-C9332D-GX2B platform switches.
- Beginning with Cisco NX-OS Release 10.1(2), Layer 2 Interfaces are supported on Cisco Nexus N9K-X9624D-R2 line cards.
- For Cisco Nexus Release 9.3(x) the Cisco Nexus N9K-C9316D-GX, N9K-C93600CD-GX switches have the following guidelines and limitations:
  - Beginning with Cisco Nexus NX-OS Release 10.1(2) Auto negotiation is supported for Speed 40G and 100G on NX-OS N9K-C93600CD-GX, N9K-C9316D-GX and N9K-C9364C-GX
  - Cisco Nexus 9300-GX platform switches do not support FC-FEC on the second lane of the 50Gx2 breakout port. The second breakout port will not link up when 50Gx2 breakout is configured. Workaround: Configure RS-FEC with 50Gx2 breakout.
  - For N9K-C9316D-GX: Ports 1-16 support 400G/100G/40G and 10G with QSA.
  - For N9K-C93600CD-GX: For ports 1-24, every four ports (1-4, 5-8, 9-12, and so on, referred to as a "quad") operate at the same speed. All the ports in a quad operate in 10G, or 40G or 100G. Mixed speed is not supported within the same quad. With QSA, all ports in a quad can operate at 10G speed. Port 25-26 should operate at same speed and port 27-28 should operate at same speed. Mismatch of speed on ports 25-26 or 27-28 is not supported.

N9K-C9364C-GX has the following guidelines and limitations:

- For ports 1-64, every four ports (1-4, 5-8, 9-12, and so on, referred to as a "quad") operates at same speed. All the ports in a quad operate in 10G, or 40G or 100G.
- Mixed speed is not supported within the same quad.
- With QSA all ports in a quad can operate at 10G speed.

Note In Cisco NX-OS Release 10.2(2)F, the link up time of SFP-10G-T-X module in N9K-C93180YC-FX3S, N9K-C93180YC-FX3 switches is 13 seconds

Default Settings for Layer 2 Interfaces

The following table lists the default settings for device access and trunk port mode parameters.

Configuring Access and Trunk Interfaces

Note If you are familiar with the Cisco IOS CLI, be aware that the Cisco NX-OS commands for this feature might differ from the Cisco IOS commands that you would use.

Guidelines for Configuring Access and Trunk Interfaces

All VLANs on a trunk must be in the same VDC.

Configuring a VLAN Interface as a Layer 2 Access Port

You can configure a Layer 2 port as an access port. An access port transmits packets on only one, untagged VLAN. You specify which VLAN traffic that the interface carries, which becomes the access VLAN. If you do not specify a VLAN for an access port, that interface carries traffic only on the default VLAN. The default VLAN is VLAN1.

The VLAN must exist before you can specify that VLAN as an access VLAN. The system shuts down an access port that is assigned to an access VLAN that does not exist.

Before you begin

Ensure that you are configuring a Layer 2 interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** {{*type slot/port*} | {**port-channel** *number*}}
3. **switchport mode** [**access** | **trunk**]
4. **switchport access vlan** *vlan-id*
5. **exit**
6. **show interface**
7. **no shutdown**
8. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** {{*type slot/port*} | {**port-channel** *number*}}
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an interface to configure, and enters interface configuration mode.

Step 3	**switchport mode** [**access** | **trunk**] **Example:** `switch(config-if)# switchport mode access`	Sets the interface as a nontrunking nontagged, single-VLAN Layer 2 interface. An access port can carry traffic in one VLAN only. By default, an access port carries traffic for

	Command or Action	Purpose
		VLAN1; to set the access port to carry traffic for a different VLAN, use the **switchport access vlan** command.
Step 4	**switchport access vlan** *vlan-id* **Example:** `switch(config-if)# switchport access vlan 5`	Specifies the VLAN for which this access port will carry traffic. If you do not enter this command, the access port carries traffic on VLAN1 only; use this command to change the VLAN for which the access port carries traffic.

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface configuration mode.

Step 6	**show interface** **Example:** `switch# show interface`	(Optional) Displays the interface status and information.

Step 7	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 8	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set Ethernet 3/1 as a Layer 2 access port that carries traffic for VLAN 5 only:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# switchport mode access
switch(config-if)# switchport access vlan 5
switch(config-if)#
```

Configuring Access Host Ports

Note You should apply the switchport host command only to interfaces that are connected to an end station.

You can optimize the performance of access ports that are connected to end stations by simultaneously setting that port as an access port. An access host port handles the STP like an edge port and immediately moves to

the forwarding state without passing through the blocking and learning states. Configuring an interface as an access host port also disables port channeling on that interface.

Note See "Configuring Port Channels" section and the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide for information about port-channel interfaces

Before you begin

Ensure that you are configuring the correct interface to an interface that is an end station.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *type slot/port*
3. **switchport host**
4. **exit**
5. **show interface**
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *type slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an interface to configure, and enters interface configuration mode.

Step 3	**switchport host** **Example:** switch(config-if)# **switchport host**	Sets the interface to be an access host port, which immediately moves to the spanning tree forwarding state and disables port channeling on this interface. **Note** Apply this command only to end stations.

Step 4	**exit**
	Example:
	`switch(config-if-range)# exit`
	`switch(config)#`
		Exits the interface mode.

	Command or Action	Purpose
Step 5	**show interface** **Example:** `switch# show interface`	(Optional) Displays the interface status and information.

Step 6	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set Ethernet 3/1 as a Layer 2 access port with PortFast enabled and port channel disabled:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# switchport host
switch(config-if)#
```

Configuring Trunk Ports

You can configure a Layer 2 port as a trunk port. A trunk port transmits untagged packets for one VLAN plus encapsulated, tagged, packets for multiple VLANs. (See the "IEEE 802.1Q Encapsulation" section for information about encapsulation.)

Note The device supports 802.1Q encapsulation only.

Before you begin

Before you configure a trunk port, ensure that you are configuring a Layer 2 interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface** {*type slot/port* | **port-channel** *number*}
3. **switchport mode** [**access** | **trunk**]
4. **exit**
5. **show interface**
6. **no shutdown**

7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** {*type slot/port* | **port-channel** *number*}
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an interface to configure, and enters interface configuration mode.

Step 3	**switchport mode** [**access** | **trunk**] **Example:** `switch(config-if)# switchport mode trunk`	Sets the interface as a Layer 2 trunk port. A trunk port can carry traffic in one or more VLANs on the same physical link (VLANs are based on the trunk-allowed VLANs list). By default, a trunk interface can carry traffic for all VLANs. To specify that only certain VLANs are allowed on the specified trunk, use the **switchport trunk allowed vlan** command.

Step 4	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 5	**show interface** **Example:** `switch# show interface`	(Optional) Displays the interface status and information.

Step 6	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set Ethernet 3/1 as a Layer 2 trunk port:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# switchport mode trunk
switch(config-if)#
```

Configuring the Allowed VLANs for Trunking Ports

You can specify the IDs for the VLANs that are allowed on the specific trunk port.

Note The **switchport trunk allowed vlan** *vlan-list* command replaces the current VLAN list on the specified port with the new list. You are prompted for confirmation before the new list is applied.

If you are doing a copy and paste of a large configuration, you might see some failures because the CLI is waiting for a confirmation before accepting other commands. To avoid this problem, you can disable prompting by using the **terminal dont-ask** command before you paste the configuration.

Before you begin

Before you configure the allowed VLANs for the specified trunk ports, ensure that you are configuring the correct interfaces and that the interfaces are trunks.

Note You can change the block of VLANs reserved for internal use. See the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide for more information about changing the reserved VLANs.

SUMMARY STEPS

1. **configure terminal**
2. **interface** {**ethernet** *slot/port* | **port-channel** *number*}
3. **switchport trunk allowed vlan** {*vlan-list* **add** *vlan-list* | **all** | **except** *vlan-list* | **none** | **remove** *vlan-list*}
4. **exit**
5. **show vlan**
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** {**ethernet** *slot/port* | **port-channel** *number*} **Example:** `switch(config)# interface ethernet 3/1`	Specifies an interface to configure, and enters interface configuration mode.
Step 3	**switchport trunk allowed vlan** {*vlan-list* **add** *vlan-list* | **all** | **except** *vlan-list* | **none** | **remove** *vlan-list*} **Example:** `switch(config-if)# switchport trunk allowed vlan add 15-20`	Sets the allowed VLANs for the trunk interface. The default is to allow all VLANs on the trunk interface: 1 to 3967 and 4048 to 4094. VLANs 3968 to 4047 are the default VLANs reserved for internal use by default. By default, all VLANs are allowed on all trunk interfaces. The default reserved VLANs are 3968 to 4094, and you can change the block of reserved VLANs. See the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide for more information. **Note** You cannot add internally allocated VLANs as allowed VLANs on trunk ports. The system returns a message if you attempt to list an internally allocated VLAN as an allowed VLAN.

Step 4	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 5	**show vlan** **Example:** `switch# show vlan`	(Optional) Displays the status and information for VLANs.

Step 6	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:**	(Optional) Copies the running configuration to the startup configuration.

	Command or Action	Purpose
	switch(config)# **copy running-config startup-config**	

Example

This example shows how to add VLANs 15 to 20 to the list of allowed VLANs on the Ethernet 3/1, Layer 2 trunk port:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# switchport trunk allowed vlan 15-20
switch(config-if)#
```

Configuring MAC Addresses Limitation on a Port

Beginning Cisco NX-OS Release 9.2(3), Cisco Nexus 9500 Series switches with N9K-X9636C-RX, N3K-C3636C-R and N3K-C36180YC-R line cards provides the ability to set an upper limit for the number of MAC addresses that can be learnt by each port. For example, if the specified VLAN limitation is 2000 MACs, the Layer 2 Forwarding Manager (L2FM) accepts the first 2000 MACs it receives and reject the remaining MACs. To configure MAC address limitation on an interface, follow these steps:

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **mac address-table limit interface port-channel** *value*
3. switch(config)# **show mac address-table limit interf**
4. switch(config)# **exit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **mac address-table limit interface port-channel** *value*	Specifies an upper limit for MAC learning at port level.
Step 3	switch(config)# **show mac address-table limit interf**	Displays the list of interfaces on which the MAC limits are configured.
Step 4	switch(config)# **exit**	Exits configuration mode.

Example

This example shows how to configure the upper limit for MAC learning at port levels:

```
switch# configure terminal
switch(config)# mac address-table limit interface port-channel 2 1000
Configuring Mac address limit will result in flushing existing Macs in the specified
VLAN/System.Proceed(yes/no)? [no]  yes
switch(config)# exit
```

This example shows how to display the MAC address limitations:

```
switch# configure terminal
switch(config)# show mac address-table limit interf
Interface    Conf Limit     Curr Count    Cfg Action    Currently
-----------    ------------   ---------     ---------    --------
Vlan1              196000             0           Flood         Flooding Unknown SA
Vlan341            196000             0           Flood         Flooding Unknown SA
Vlan342            196000             0           Flood         Flooding Unknown SA
Vlan343            196000             0           Flood         Flooding Unknown SA
Vlan344            196000             0           Flood         Flooding Unknown SA
Vlan345            196000             0           Flood         Flooding Unknown SA
Vlan346            196000             0           Flood         Flooding Unknown SA
Vlan347            196000             0           Flood         Flooding Unknown SA
Vlan348            196000             0           Flood         Flooding Unknown SA
Vlan349            196000             0           Flood         Flooding Unknown SA
Vlan350            196000             0           Flood         Flooding Unknown SA
port-channel1      196000             0           Flood         Flooding Unknown SA
port-channel2      1000               0           Flood         Flooding Unknown SA
port-channel11     196000             0           Flood         Flooding Unknown SA
port-channel12     196000             0           Flood         Flooding Unknown SA
port-channel13     196000             0           Flood         Flooding Unknown SA
port-channel601    196000             0           Flood         Flooding Unknown SA
port-channel603    196000             0           Flood         Flooding Unknown SA
port-channel888    196000             0           Flood         Flooding Unknown SA
Ethernet1/6        196000             0           Flood         Flooding Unknown SA
Ethernet1/15       196000             0           Flood         Flooding Unknown SA
Ethernet1/35       196000             0           Flood         Flooding Unknown SA
BF2(config)#
switch(config)# exit
```

Configuring a Default Interface

The default interface feature allows you to clear the existing configuration of multiple interfaces such as Ethernet, loopback, VLAN network, port-channel, and tunnel interfaces. All user configuration under a specified interface will be deleted. You can optionally create a checkpoint before clearing the interface configuration so that you can later restore the deleted configuration.

Note The default interface feature is not supported for management interfaces because the device could go to an unreachable state.

If the speed group is configured, the **default interface** command displays the following error:

```
Error: default interface is not supported as speed-group is configured
```

SUMMARY STEPS

1. **configure terminal**
2. **default interface** *int-if* [**checkpoint** *name*]

3. **exit**
4. **show interface**
5. **no shutdown**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**default interface** *int-if* [**checkpoint** *name*] **Example:** `switch(config)# default interface ethernet 3/1 checkpoint test8`	Deletes the configuration of the interface and restores the default configuration. Use the **?** keyword to display the supported interfaces. Use the **checkpoint** keyword to store a copy of the running configuration of the interface before clearing the configuration.

Step 3	**exit**
	Example:
	`switch(config)# exit`
	`switch(config)#`
		Exits global configuration mode.

Step 4	**show interface** **Example:** `switch# show interface`	(Optional) Displays the interface status and information.

Step 5	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Example

This example shows how to delete the configuration of an Ethernet interface while saving a checkpoint of the running configuration for rollback purposes:

```
switch# configure terminal
switch(config)# default interface ethernet 3/1 checkpoint test8
.......Done
switch(config)#
```

Configuring SVI Autostate Disable for the System

You can manage an SVI with the SVI autostate feature. You can configure the SVI autostate disable feature to keep an SVI up even if no interface is up in the corresponding VLAN. (Similarly, configure the SVI autostate enable feature so an SVI goes down when no interface is up in the corresponding VLAN). Use this procedure to configure this feature for the entire system.

Note The **system default interface-vlan autostate** command enables the SVI autostate feature.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **system default interface-vlan autostate**
3. **no shutdown**
4. **show running-config** [**all**]

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	[**no**] **system default interface-vlan autostate** **Example:** `switch(config)# no system default interface-vlan autostate`	Disables the default autostate behavior for the device. **Note** Use the **system default interface-vlan autostate** command to enable the autostate behavior for the device.

Step 3	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 4	**show running-config** [**all**] **Example:** `switch(config)# show running-config`	(Optional) Displays the running configuration. To display the default and configured information, use the **all** keyword.

Example

This example shows how to disable the default autostate behavior on the Cisco NX-OS device:

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

Configuring SVI Autostate Disable Per SVI

You can configure SVI autostate enable or disable on individual SVIs. The SVI-level setting overrides the system-level SVI autostate configuration for that particular SVI.

SUMMARY STEPS

1. **configure terminal**
2. **feature interface-vlan**
3. **interface vlan** *vlan-id*
4. [**no**] **autostate**
5. **exit**
6. **show running-config interface vlan** *vlan-id*
7. **no shutdown**
8. **show startup-config interface vlan** *vlan-id*

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature interface-vlan** **Example:** `switch(config)# feature interface-vlan`	Enables VLAN interface mode.

Step 3	**interface vlan** *vlan-id*
	Example:
	`switch(config-if)# interface vlan10`
	`switch(config)#`
		Creates a VLAN interface and enters interface configuration mode. The range is from 1 and 4094.

Step 4	[**no**] **autostate** **Example:** `switch(config-if)# no autostate`	By default, enables the SVI autostate feature on specified interface. To disable the default settings, use the **no** form of this command.
Step 5	**exit** **Example:**	Exits the interface configuration mode.

	Command or Action	Purpose
	`switch(config-if)# exit` `switch(config)#`	
Step 6	**show running-config interface vlan** *vlan-id* **Example:** `switch(config)# show running-config interface vlan10`	(Optional) Displays the running configuration for the specified VLAN interface.

Step 7	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 8	**show startup-config interface vlan** *vlan-id* **Example:** `switch(config)# show startup-config interface vlan10`	(Optional) Displays the VLAN configuration in the startup configuration.

Example

This example shows how to disable the default autostate behavior on an individual SVI:

```
switch# configure terminal
switch(config)# feature interface-vlan
switch(config)# interface vlan10
witch(config-if)# no autostate
```

Configuring the Device to Tag Native VLAN Traffic

When you are working with 802.1Q trunked interfaces, you can maintain the tagging for all packets that enter with a tag that matches the value of the native VLAN ID and drops all untagged traffic (you will still carry control traffic on that interface). This feature applies to the entire device; you cannot apply it to selected VLANs on a device.

The **vlan dot1q tag native global** command changes the behavior of all native VLAN ID interfaces on all trunks on the device.

Note If you enable 802.1Q tagging on one device and disable it on another device, all traffic is dropped on the device and this feature is disabled. You must configure this feature identically on each device.

SUMMARY STEPS

1. **configure terminal**
2. **vlan dot1q tag native**
3. **exit**

4. **show vlan**
5. **no shutdown**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vlan dot1q tag native** **Example:** `switch(config)# vlan dot1q tag native`	Modifies the behavior of a 802.1Q trunked native VLAN ID interface. The interface maintains the taggings for all packets that enter with a tag that matches the value of the native VLAN ID and drops all untagged traffic. The control traffic is still carried on the native VLAN.

Step 3	**exit**
	Example:
	`switch(config-if-range)# exit`
	`switch(config)#`
		Exits the interface configuration mode.

Step 4	**show vlan** **Example:** `switch# show vlan`	(Optional) Displays the status and information for VLANs.

Step 5	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to change the behavior of the native VLAN on an 802.1Q trunked interface to maintain the tagged packets and drop all untagged traffic (except control traffic):

```
switch# configure terminal
switch(config)# vlan dot1q tag native
switch#
```

Configuring Interface Breakout Profile for 50-G Interfaces in a 16-Slot Chassis

The interface breakout profile is needed to breakout high bandwidth 100-G ports into two 50-G interfaces for slot 8 to 16 in the Cisco Nexus 9516 switch for -EX line cards.

SUMMARY STEPS

1. **configure terminal**
2. (Optional) **interface breakout-profile 50g-2x-only**
3. **copy running-config startup-config**
4. **reload**
5. **interface breakout module** *module-number* **port** *port-range* **map [10g-4x | 25g-4x | 50g-2x]**

DETAILED STEPS

Procedure

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

Step 2	(Optional) **interface breakout-profile 50g-2x-only**
	Example:
	`switch(config)# interface breakout-profile 50g-2x-only`
	`Warning: Please save config and reload the switch for breakout-profile config to take effect`
	`Please save config and reload the switch for the configuration to take effect`
		This command is required to breakout slots 8 to 16. It is not required for slots 1 to 7.

Step 3	**copy running-config startup-config**
	Example:
	`switch(config-inf)# copy running-config startup-config`
	`[########################################] 100%`
	`Copy complete, now saving to disk (please wait)...`
	`Copy complete.`
		Copies the running configuration to the startup configuration.

Step 4	**reload**
	Example:
	`switch(config-inf)# reload`
	`This command will reboot the system. (y/n)? [n] y`
		Reboots the switch.
		Note
		After the switch reloads and the modules are up, enter the following CLI for any module or ports to breakout.

Step 5	**interface breakout module** *module-number* **port** *port-range* **map [10g-4x | 25g-4x | 50g-2x]** **Example:**	Breaks out the 100-Gb port to 2 50-Gb ports. The range of *module-number* is 1 to 30. The range of *port-range* is 1 to 72.

	Command or Action	Purpose
	`switch(config)# interface breakout module 1 port 1-32 map 50g-2x`	

Changing the System Default Port Mode to Layer 2

You can set the system default port mode to Layer 2 access ports.

SUMMARY STEPS

1. **configure terminal**
2. **system default switchport** [**shutdown**]
3. **exit**
4. **show interface brief**
5. **no shutdown**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**system default switchport** [**shutdown**] **Example:** switch(config-if)# **system default switchport**	Sets the default port mode for all interfaces on the system to Layer 2 access port mode and enters interface configuration mode. By default, all the interfaces are Layer 3. **Note** When the **system default switchport shutdown** command is issued: • Any FEX HIFs that are not configured with **no shutdown** are shutdown. To avoid the shutdown, configure the FEX HIFs with **no shut** • Any Layer 2 port that is not specifically configured with **no shutdown** are shutdown. To avoid the shutdown, configure the Layer 2 port with **no shut**

Step 3	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface configuration mode.

	Command or Action	Purpose
Step 4	**show interface brief** **Example:** `switch# show interface brief`	(Optional) Displays the status and information for interfaces.

Step 5	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set the system ports to be Layer 2 access ports by default:

```
switch# configure terminal
switch(config-if)# system default switchport
switch(config-if)#
```

Verifying the Interface Configuration

To display access and trunk interface configuration information, perform one of the following tasks.

Command	Purpose
show interface ethernet *slot/port* [**brief** | | **counters** | **debounce** | **description** | **flowcontrol** | **mac-address** | **status** | **transceiver**]	Displays the interface configuration.
show interface brief	Displays interface configuration information, including the mode.
show interface switchport	Displays information, including access and trunk interface, information for all Layer 2 interfaces.
show interface trunk [**module** *module-number* | **vlan** *vlan-id*]	Displays trunk configuration information.
show interface capabilities	Displays information about the capabilities of the interfaces.

Command	Purpose
show running-config [**all**]	Displays information about the current configuration. The **all** command displays the default and current configurations.
show running-config interface ethernet *slot/port*	Displays configuration information about the specified interface.
show running-config interface port-channel *slot/port*	Displays configuration information about the specified port-channel interface.
show running-config interface vlan *vlan-id*	Displays configuration information about the specified VLAN interface.

Monitoring the Layer 2 Interfaces

Use the following commands to display Layer 2 interfaces:

Command	Purpose
clear counters interface [**interface**]	Clears the counters.
load- interval {**interval** *seconds* {**1** | **2** | **3**}}	Cisco Nexus 9000 Series devices set three different sampling intervals to bit-rate and packet-rate statistics.
show interface counters [**module** *module*]	Displays input and output octets unicast packets, multicast packets, and broadcast packets.
show interface counters detailed [**all**]	Displays input packets, bytes, and multicast as well as output packets and bytes.
show interface counters errors [**module** *module*]	Displays information on the number of error packets.

Configuration Examples for Access and Trunk Ports

This example shows how to configure a Layer 2 access interface and assign the access VLAN mode for that interface:

```
switch# configure terminal
switch(config)# interface ethernet 2/30
switch(config-if)# switchport
switch(config-if)# switchport mode access
switch(config-if)# switchport access vlan 5
switch(config-if)#
```

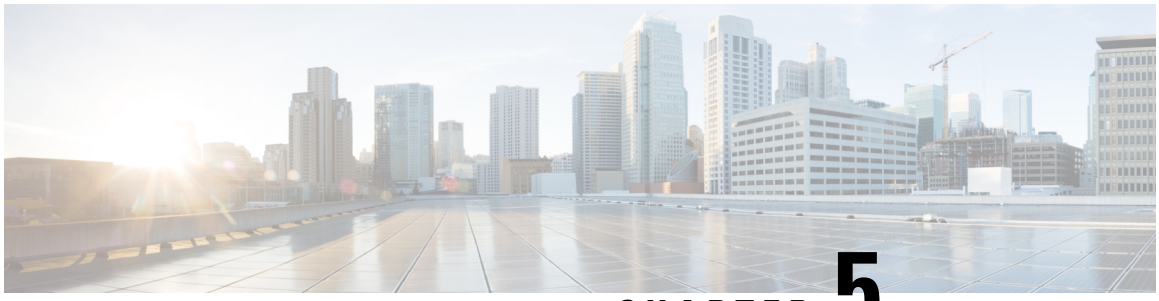
This example shows how to configure a Layer 2 trunk interface, assign the native VLAN and the allowed VLANs, and configure the device to tag the native VLAN traffic on the trunk interface:

```
switch# configure terminal
switch(config)# interface ethernet 2/35
switch(config-if)# switchport
```

```
switch(config-if)# switchport mode trunk
switch(config-if)# switchport trunk native vlan 10
switch(config-if)# switchport trunk allowed vlan 5, 10
switch(config-if)# exit
switch(config)# vlan dot1q tag native
switch(config)#
```

Related Documents

Related Documents	Document Title
Configuring Layer 3 interfaces	Configuring Layer 2 Interfaces section
Port Channels	Configuring Port Channels section
VLANs, private VLANs, and STP	*Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide*
System management	*Cisco Nexus 9000 Series NX-OS System Management Configuration Guide*
High availability	*Cisco Nexus 9000 Series NX-OS High Availability and Redundancy Guide*
Licensing	*Cisco NX-OS Licensing Guide*
Release Notes	*Cisco Nexus 9000 Series NX-OS Release Notes*



CHAPTER 5

Configuring Layer 3 Interfaces

About Layer 3 Interfaces

Layer 3 interfaces forward IPv4 and IPv6 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 routed interfaces by default. You can change this default behavior with the CLI setup script.

Note The default behavior varies based on the type of switch (Cisco Nexus 9300, Cisco Nexus 9500, or Cisco Nexus 3164).

Note Cisco Nexus 9300 Series switches (except Cisco Nexus 9332 switch) have a Layer 2 default mode.

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

You can also create a Layer 3 port channel from routed interfaces. For more information about port channels, see the “Configuring Port Channels” section.

Routed interfaces 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.

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.

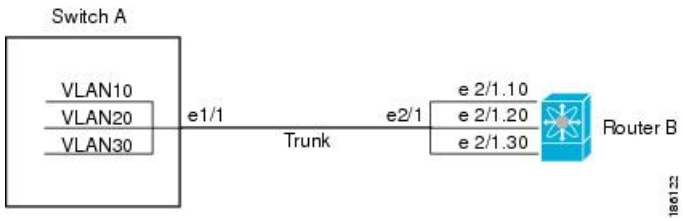
You create a subinterface with a name that consists of the parent interface name (for example, Ethernet 2/1) followed by a period and then by a number that is unique for that subinterface. For example, you could create a subinterface for Ethernet interface 2/1 named Ethernet 2/1.1 where .1 indicates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

Figure 4: Subinterfaces for VLANs



For more information about VLANs, see the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide.

VLAN Interfaces

A VLAN interface, or 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.

However, 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.

Enable the VLAN network interface feature using the **feature interface-vlan** configuration. The system automatically takes a checkpoint prior to disabling the feature, and you can roll back to this checkpoint. See the Cisco Nexus 9000 Series NX-OS System Management Configuration Guide for information on rollbacks and checkpoints.

Note The **feature interface-vlan** configuration is not available on the Nexus 9800 switches.

Layer 3 inter-VLAN Routing

You can route traffic across VLAN interfaces to provide Layer 3 inter-VLAN routing by configuring a VLAN interface for each VLAN, and assigning an IP address on the VLAN interface.

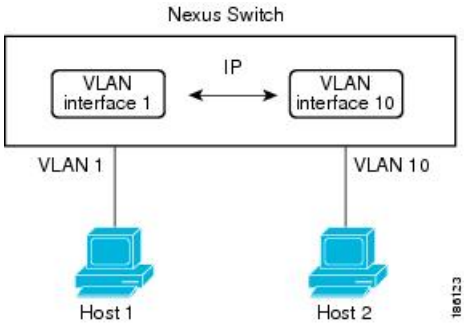
For more information about IP addresses and IP routing, see the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide.

Connecting Two VLAN Interfaces

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

The following figure shows two hosts connected to two VLANs on a device.

Figure 5: Connecting Two VLANs with VLAN interfaces



Note You cannot delete the VLAN interface for VLAN 1.

Loopback Interfaces

A loopback interface is a virtual interface with a single endpoint that is always up. Any packet transmitted over a loopback interface is immediately received by this interface. Loopback interfaces emulate a physical interface. You can configure up to 1024 loopback interfaces, numbered 0 to 1023.

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.

High Availability

Layer 3 interfaces support stateful and stateless restarts. After the switchover, Cisco NX-OS applies the runtime configuration after the switchover.

See the Cisco Nexus 9000 Series NX-OS High Availability and Redundancy Guide for complete information about high availability.

Virtualization Support

Layer 3 interfaces support Virtual Routing and Forwarding instances (VRFs). VRFs exist within virtual device contexts (VDCs). By default, Cisco NX-OS places you in the default VDC and default VRF .

Note You must assign an interface to a VRF before you configure the IP address for that interface.

Layer 3 Static MAC Addresses

You can configure a static MAC address for the following Layer 3 interfaces:

- Layer 3 interfaces
- Layer 3 subinterfaces
- Layer 3 port channels
- VLAN network interface

Note You cannot configure static MAC address on tunnel interfaces.

Prerequisites for Layer 3 Interfaces

Layer 3 interfaces have the following prerequisites:

- You are familiar with IP addressing and basic configuration. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information about IP addressing.

Guidelines and Limitations for Layer 3 Interfaces

Layer 3 interfaces have the following configuration guidelines and limitations:

- **show** commands with the **internal** keyword are not supported.
- Configuring a subinterface on a physical interface that is configured to be a member of a port-channel is not supported. One must configure the subinterface under the port-channel interface itself.
- Assigning a user-defined random MAC address to a Layer 3 interface results in silent dropping of transit traffic through that interface. This issue occurs on these switches:
  - N9K-X9636C-R
  - N9K-X9636Q-R
  - N9K-X9636C-RX
  - N9K-X96136YC-R (48X1G/10G/25G and 6X40G/100G)
  - N3K-C36180YC-R
  - N3K-C3636C-R

  To restore normal traffic flow, assign only one unique MAC address per Layer 3 logical interface that matches the 36-bit MSB of the MAC address.
- The Dynamic Host Configuration Protocol (DHCP) option is not supported when configuring a subinterface on a port-channel interface.
- IPv6 counters for SVI and subinterfaces on Cisco Nexus 9500 Series Switches with X9700-EX and X9700-FX line cards are not supported.
- Multicast and/or broadcast counters for both SVI and subinterfaces are not supported.
- Control plane SVI/SI traffic for both SVI and subinterfaces counters are not supported.
- Beginning Cisco NX-OS Release 9.3(6), sub-interface multicast and broadcast counters are supported on Cisco Nexus N9K-C9336C-FX2 and N9K-C93240YC-FX2 switches.
- The SVI, Layer 2 VLAN, MPLS counters may not work when you enable subinterface multicast and broadcast counters.
- Up to 1000 subinterfaces are supported for this statistics.
- Beginning with Cisco NX-OS Release 10.2(1q)F, Layer 3 (L3) interfaces are supported on the N9K-C9332D-GX2B platform switches.
- Beginning with Cisco NX-OS Release 10.1(2), Layer 3 Interfaces are supported on Cisco Nexus N9K-X9624D-R2 line card.
- Beginning with Cisco NX-OS Release 10.3(1)F, the Cisco Nexus 9808 platform switches support L3, Loopback, and Subinterfaces.
- Beginning with Cisco NX-OS Release 10.3(1)F, the statistics support is provided for L3 Physical and Subinterface on Cisco Nexus 9808 platform switches.

- Cisco Nexus 9800 platform switches have the following limitations for L3 Physical and Subinterface support:
  - Broadcast counters is not supported.
  - **hardware profile sub-interface flex-stats** command is not applicable.
  - Subinterface statistics are not aggregated to parent interface.

Note If you are familiar with the Cisco IOS CLI, be aware that the Cisco NX-OS commands for this feature might differ from the Cisco IOS commands that you would use.

Default Settings

The following table lists the default settings for Layer 3 interface parameters.

Table 10: Default Layer 3 Interface Parameters

Parameters	Default
Admin state	Shut

Configuring Layer 3 Interfaces

Configuring a Routed Interface

You can configure any Ethernet port as a routed interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port*
3. **no switchport**
4. [**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
5. **show interfaces**
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 interface.

Step 4	[**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
	Example:
	`switch(config-if)# ip address 192.0.2.1/8`
	Example:
	`switch(config-if)# ipv6 address 2001:0DB8::1/8`
		• Configures an IP address for this interface. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information about IP addresses.
		• Configures an IPv6 address for this interface. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information about IPv6 addresses.

Step 5	**show interfaces** **Example:** `switch(config-if)# show interfaces ethernet 2/1`	(Optional) Displays the Layer 3 interface statistics.

Step 6	**no shutdown**
	Example:
	`switch#`
	`switch(config-if)# int e2/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves the configuration change.

Example

- Use the **medium** command to set the interface medium to either point to point or broadcast.

Command	Purpose
medium {**broadcast** | **p2p**} Example: `switch(config-if)# medium p2p`	Configures the interface medium as either point to point or broadcast.

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.

- Use the **switchport** command to convert a Layer 3 interface into a Layer 2 interface.

Command	Purpose
switchport Example: `switch(config-if)# switchport`	Configures the interface as a Layer 2 interface and deletes any configuration specific to Layer 3 on this interface.

- This example shows how to configure a routed 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
```

The default setting for interfaces is routed. If you want to configure an interface for Layer 2, enter the **switchport** command. Then, if you change a Layer 2 interface to a routed interface, enter the **no switchport** command.

Configuring a Subinterface on a Routed Interface

You can configure one or more subinterfaces on a routed interface made from routed interfaces.

Before you begin

Configure the parent interface as a routed interface.

See the “Configuring a Routed Interface” section.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *slot/port.number*
3. [**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
4. **encapsulation dot1Q** *vlan-id*
5. **show interfaces**

6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface ethernet** *slot/port.number*
	Example:
	`switch(config)# interface ethernet 2/1.1`
	`switch(config-subif)#`
		Creates a subinterface and enters subinterface configuration mode. The number range is from 1 to 4094.

Step 3	[**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
	Example:
	`switch(config-subif)# ip address 192.0.2.1/8`
	Example:
	`switch(config-subif)# ipv6 address 2001:0DB8::1/8`
		• Configures an IP address for this subinterface. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information on IP addresses.
		• Configures an IPv6 address for this subinterface. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information on IPv6 addresses.

Step 4	**encapsulation dot1Q** *vlan-id* **Example:** `switch(config-subif)# encapsulation dot1Q 33`	Configures IEEE 802.1Q VLAN encapsulation on the subinterface. The range is from 2 to 4093.
Step 5	**show interfaces** **Example:** `switch(config-subif)# show interfaces ethernet 2/1.1`	(Optional) Displays the Layer 3 interface statistics.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves the configuration change.

Example

- This example shows how to create a subinterface:

```
switch# configure terminal
switch(config)# interface ethernet 2/1.1
switch(config-if)# ip address 192.0.2.1/8
```

```
switch(config-if)# encapsulation dot1Q 33
switch(config-if)# copy running-config startup-config
```

- The output of the **show interface eth** command is enhanced for the subinterfaces as shown in the following :

```
switch# show interface ethernet 1/2.1
Ethernet1/2.1 is down (Parent Interface Admin down)
admin state is down, Dedicated Interface, [parent interface is Ethernet1/2]
Hardware: 40000 Ethernet, address: 0023.ac67.9bc1 (bia 4055.3926.61d4)
Internet Address is 10.10.10.1/24
MTU 1500 bytes, BW 40000000 Kbit, DLY 10 usec
reliability 255/255, txload 1/255, rxload 1/255
Auto-mdix is turned off
EtherType is 0x8100
L3 in Switched:
 ucast: 0 pkts, 0 bytes - mcast: 0 pkts, 0 bytes
L3 out Switched:
    ucast: 0 pkts, 0 bytes - mcast: 0 pkts, 0 bytes
```

Configuring a VLAN Interface

You can create VLAN interfaces to provide inter-VLAN routing.

SUMMARY STEPS

1. **configure terminal**
2. **feature interface-vlan**
3. **interface vlan** *number*
4. [**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
5. **show interface vlan** *number*
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**feature interface-vlan** **Example:** `switch(config)# feature interface-vlan`	Enables VLAN interface mode.
Step 3	**interface vlan** *number* **Example:**	Creates a VLAN interface. The number range is from 1 to 4094.

	Command or Action	Purpose
	`switch(config)# interface vlan 10` `switch(config-if)#`	

Step 4	[**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
	Example:
	`switch(config-if)# ip address 192.0.2.1/8`
	Example:
	`switch(config-if)# ipv6 address 2001:0DB8::1/8`
		• Configures an IP address for this VLAN interface. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information on IP addresses.
		• Configures an IPv6 address for this VLAN interface. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information on IPv6 addresses.

Step 5	**show interface vlan** *number* **Example:** `switch(config-if)# show interface vlan 10`	(Optional) Displays the Layer 3 interface statistics.

Step 6	**no shutdown**
	Example:
	`switch(config)# int e3/1`
	`switch(config)# no shutdown`
		(Optional) Clears the errors on the interfaces where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

Example

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 Static MAC Address on a Layer 3 Interface

You can configure static MAC addresses on Layer 3 interfaces. You cannot configure broadcast or multicast addresses as static MAC addresses.

Note You cannot configure static MAC addresses on tunnel interfaces.

Note This configuration is limited to 16 VLAN interfaces. Applying the configuration to additional VLAN interfaces results in a down state for the interface with a `Hardware prog failed.` status.

SUMMARY STEPS

1. **config t**
2. **interface** [**ethernet** *slot/port* | **ethernet** *slot/port.number* | **port-channel** *number* | **vlan** *vlan-id*]
3. **mac-address** *mac-address*
4. **exit**
5. (Optional) **show interface** [**ethernet** *slot/port* | **ethernet** *slot/port.number* | **port-channel** *number* | **vlan** *vlan-id*]
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**config t**
	Example:
	`switch# config t`
	`switch(config)#`
		Enters configuration mode.

Step 2	**interface** [**ethernet** *slot/port* | **ethernet** *slot/port.number* | **port-channel** *number* | **vlan** *vlan-id*] **Example:** `switch(config)# interface ethernet 7/3`	Specifies the Layer 3 interface and enters the interface configuration mode. **Note** You must create the Layer 3 interface before you can assign the static MAC address.

Step 3	**mac-address** *mac-address*
	Example:
	`switch(config-if)# mac-address 22ab.47dd.ff89`
	`switch(config-if)#`
		Specified a static MAC address to add to the Layer 3 interface.

Step 4	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode.

Step 5	(Optional) **show interface** [**ethernet** *slot/port* | **ethernet** *slot/port.number* | **port-channel** *number* | **vlan** *vlan-id*] **Example:** `switch# show interface ethernet 7/3`	Displays information about the Layer 3 interface.

	Command or Action	Purpose
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Example

This example shows how to configure the Layer 3 interface on slot 7, port 3 with a static MAC address:

```
switch# config t
switch(config)# interface ethernet 7/3
switch(config-if)# mac-address 22ab.47dd.ff89
switch(config-if)#
```

Configuring a Loopback Interface

You can configure a loopback interface to create a virtual interface that is always up.

Before you begin

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

SUMMARY STEPS

1. **configure terminal**
2. **interface loopback** *instance*
3. [**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
4. **show interface loopback** *instance*
5. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**interface loopback** *instance*
	Example:
	`switch(config)# interface loopback 0`
	`switch(config-if)#`
		Creates a loopback interface. The range is from 0 to 1023.

	Command or Action	Purpose

Step 3	[**ip address** *ip-address/length* | **ipv6 address** *ipv6-address/length*]
	Example:
	`switch(config-if)# ip address 192.0.2.1/8`
	Example:
	`switch(config-if)# ipv6 address 2001:0DB8::1/8`
		• Configures an IP address for this interface. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information about IP addresses.
		• Configures an IPv6 address for this interface. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information about IPv6 addresses.

Step 4	**show interface loopback** *instance* **Example:** `switch(config-if)# show interface loopback 0`	(Optional) Displays the loopback interface statistics.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

Example

This example shows how to create a loopback interface:

```
switch# configure terminal
switch(config)# interface loopback 0
switch(config-if)# ip address 192.0.2.1/8
switch(config-if)# copy running-config startup-config
```

Configuring PBR on SVI on the Gateway

This procedure configures PBR on the primary SVI interface in the gateway.

Note Steps 2 through 6 are needed if you want to configure a PBR policy on the unnumbered Primary/Secondary VLAN interfaces. This is not mandatory for IP unnumbered on the SVI feature.

SUMMARY STEPS

1. **configure terminal**
2. **ip access-list** *list-name*
3. **permit tcp host** *ipaddr* **host** *ipaddr* **eq** *port-number*
4. **exit**
5. **route-map** *route-map-name*
6. **match ip address** *access-list-name*
7. **set ip next-hop** *addr1*

8. **exit**
9. **interface vlan** *vlan-id*
10. **ip address** *ip-addr*
11. **no ip redirects**
12. (Optional) **ip policy route-map pbr-sample**
13. **exit**
14. **hsrp version 2**
15. **hsrp***group-num*
16. **name** *name-val*
17. **ip** *ip-addr*
18. **no shutdown**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enter global configuration mode.
Step 2	**ip access-list** *list-name* **Example:** `switch(config)# ip access-list pbr-sample`	Configure access list.
Step 3	**permit tcp host** *ipaddr* **host** *ipaddr* **eq** *port-number* **Example:** `switch(config-acl)# permit tcp host 10.1.1.1 host 192.168.2.1 eq 80`	Specify the packets to forward on a specific port.
Step 4	**exit** **Example:** `switch(config-acl)# exit`	Exit configuration mode.
Step 5	**route-map** *route-map-name* **Example:** `switch(config)# route-map pbr-sample`	Create a route-map or enter route-map command mode.
Step 6	**match ip address** *access-list-name* **Example:** `switch(config-route-map)# match ip address pbr-sample`	Match values from the routing table.
Step 7	**set ip next-hop** *addr1* **Example:**	Set IP address of the next hop.

	Command or Action	Purpose
	`switch(config-route-map)# set ip next-hop 192.168.1.1`	
Step 8	**exit** **Example:** `switch(config-route-map)# exit`	Exit command mode.
Step 9	**interface vlan** *vlan-id* **Example:** `switch(config)# interface vlan 2003`	Creates a VLAN interface and enters interface configuration mode. The range is from 1 and 4094.This is the primary VLAN.
Step 10	**ip address** *ip-addr* **Example:** `switch(config-if)# ip address 10.0.0.1/8`	Configures an IP address for the interface.
Step 11	**no ip redirects** **Example:** `switch(config-if)# no ip redirects`	Needs to be configured on all unnumbered primary and secondary VLAN interfaces.
Step 12	(Optional) **ip policy route-map pbr-sample** **Example:** `switch(config-if)# ip policy route-map pbr-sample`	Enter this command if you want to apply a PBR policy on the unnumbered Primary/Secondary VLAN interface.
Step 13	**exit** **Example:** `switch(config-if)# exit`	Exit command mode.
Step 14	**hsrp version 2** **Example:** `switch(config-if)# hsrp version 2`	Set the HSRP version.
Step 15	**hsrp***group-num* **Example:** `switch(config-if)# hsrp 200`	Set the HSRP group number.
Step 16	**name** *name-val* **Example:** `switch(config-if-hsrp)# name primary`	Configure the redundancy name string.
Step 17	**ip** *ip-addr* **Example:** `switch(config-if-hsrp)# ip 10.0.0.100`	Configures an IP address.
Step 18	**no shutdown** **Example:**	Negates shutdown.

	Command or Action	Purpose
	`switch(config-if-hsrp)# no shutdown`	

Configuring IP Unnumbered on SVI Secondary VLAN on the Gateway

This procedure configures IP unnumbered on the secondary SVI in the gateway. Beginning Cisco NX-OS Release 9.3(6), this feature is supported on Cisco Nexus N9K-C9316D-GX, N9K-C93600CD-GX, N9K-C9364C-GX switches.

SUMMARY STEPS

1. **configure terminal**
2. **interface vlan** *vlan-list*
3. **ip unnumbered vlan** *primary-vlan-id*
4. (Optional) **ip policy route-map pbr-sample**
5. **no ip redirects**
6. **hsrp version 2**
7. **hsrp** *group-num*
8. **follow** *name*
9. **ip** *ip-addr*
10. **no shutdown**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enter configuration mode.
Step 2	**interface vlan** *vlan-list* **Example:** `switch(config)# interface vlan 2001`	Creates a VLAN interface and enters interface configuration mode. The range is from 1 to 4094. This is the secondary VLAN.
Step 3	**ip unnumbered vlan** *primary-vlan-id* **Example:** `switch(config-if)# ip unnumbered vlan 2003`	Enables IP processing on an interface without assigning an explicit IP address to an interface.
Step 4	(Optional) **ip policy route-map pbr-sample** **Example:** `switch(config-if)# ip policy route-map pbr-sample`	Enter this command if you want to apply a PBR policy on the unnumbered Primary/Secondary VLAN interface.
Step 5	**no ip redirects** **Example:**	Needs to be configured on all unnumbered primary and secondary VLAN interfaces.

	Command or Action	Purpose
	`switch(config-if)# no ip redirects`	
Step 6	**hsrp version 2** **Example:** `switch(config-if)# hsrp version 2`	Set the HSRP version.
Step 7	**hsrp** *group-num* **Example:** `switch(config-if)# hsrp 200`	Set the HSRP group number.
Step 8	**follow** *name* **Example:** `switch(config-if-hsrp)# follow primary`	Configure the group to be followed.
Step 9	**ip** *ip-addr* **Example:** `switch(config-if-hsrp)# ip 10.0.0.100`	Enters HRSP IPv4 and sets the virtual IP address.
Step 10	**no shutdown** **Example:** `switch(config-if-hsrp)# no shutdown`	Negate shutdown.

Configuring SVI TCAM Region

Beginning Cisco NX-OS Release 9.3(3), you can display Layer 3 statistics on SVI interfaces on Cisco Nexus 3100 Series switches. You can change the size of the SVI ternary content addressable memory (TCAM) regions in the hardware to display the Layer 3 incoming unicast counters on SVI interfaces.

SUMMARY STEPS

1. **hardware profile tcam region** {**arpacl** | **e-racl**} | **ifacl** | **nat** | **qos**} |**qoslbl** | **racl**} | **vacl** | **svi** } *tcam_size*
2. **copy running-config startup-config**
3. switch(config)# **show hardware profile tcam region**
4. switch(config)# **reload**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**hardware profile tcam region** {**arpacl**	**e-racl**}	**ifacl**	**nat**	**qos**} |**qoslbl**	**racl**}	**vacl**	**svi** } *tcam_size*	Changes the ACL TCAM region size. • **arpacl**—Configures the size of the Address Resolution Protocol (ARP) ACL (ARPACL) TCAM region.

	Command or Action	Purpose
		• **e-racl**—Configures the size of the egress router ACL (ERACL) TCAM region. • **e-vacl**—Configures the size of the egress VLAN ACL (EVACL) TCAM region. • **ifacl**—Configures the size of the interface ACL (ifacl) TCAM region. The maximum number of entries is 1500. • **nat**—Configures the size of the NAT TCAM region. • **qos**—Configures the size of the quality of service (QoS) TCAM region. • **qoslbl**—Configures the size of the QoS Label (qoslbl) TCAM region. • **racl**—Configures the size of the router ACL (RACL) TCAM region. • **vacl**—Configures the size of the VLAN ACL (VACL) TCAM region. • *svi*—Configures the size of the SVI TCAM region. The default size of SVI TCAM size is 0. • *tcam_size*—TCAM size. The range is from 0 to 2,14,74, 83, 647 entries. **Note** **vacl** and **e-vacl** TCAM regions should be set to the same size.
Step 2	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.
Step 3	switch(config)# **show hardware profile tcam region** **Example:** `switch(config)# show hardware profile tcam region`	Displays the TCAM sizes that will be applicable on the next reload of the switch.
Step 4	switch(config)# **reload** **Example:** `switch(config)# reload`	Copies the running configuration to the startup configuration. **Note** The new size values are effective only upon the next reload after saving the **copy running-config to startup-config**.

Example

The following example shows how to change the size of the SVI TCAM region:

```
switch(config)# hardware profile tcam region svi 256
[SUCCESS] New tcam size will be applicable only at boot time.
You need to 'copy run start' and 'reload'

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
```

Assigning an Interface to a VRF

You can add a Layer 3 interface to a VRF.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type number*
3. **vrf member** *vrf-name*
4. **ip address** *ip-prefix/length*
5. **show vrf** [*vrf-name*] **interface** *interface-type number*
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**interface** *interface-type number*
	Example:
	`switch(config)# interface loopback 0`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**vrf member** *vrf-name* **Example:** `switch(config-if)# vrf member RemoteOfficeVRF`	Adds this interface to a VRF.
Step 4	**ip address** *ip-prefix/length* **Example:** `switch(config-if)# ip address 192.0.2.1/16`	Configures an IP address for this interface. You must do this step after you assign this interface to a VRF.

	Command or Action	Purpose
Step 5	**show vrf** [*vrf-name*] **interface** *interface-type number* **Example:** `switch(config-vrf)# show vrf Enterprise interface loopback 0`	(Optional) Displays VRF information.
Step 6	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

Example

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 a DHCP Client on an Interface

You can configure the DHCP client on an SVI, a management interface, or a physical Ethernet interface for IPv4 or IPv6 address

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface ethernet** *type slot/port* | **mgmt** *mgmt-interface-number* | **vlan** *vlan id*
3. switch(config-if)# [**no**] **ipv6 address use-link-local-only**
4. switch(config-if)# [**no**] [**ip** | **ipv6] address dhcp**
5. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

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. The range of *vlan id* is from 1 to 4094.
Step 3	switch(config-if)# [**no**] **ipv6 address use-link-local-only**	Prepares for request to the DHCP server.

	Command or Action	Purpose
		Note This command is only required for an IPv6 address.
Step 4	switch(config-if)# [**no**] [**ip** | **ipv6] address dhcp**	Requests the DHCP server for an IPv4 or IPv6 address. The **no** form of this command removes any address that was acquired.
Step 5	(Optional) switch(config)# **copy running-config startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

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 use-link-local-only
switch(config-if)# ipv6 address dhcp
```

Configuring SVI and Subinterface Ingress/Egress Unicast Counters

Beginning Cisco NX-OS Release 9.3(3), SVI and subinterface unicast counters are supported on Cisco Nexus 9300-EX, 9300-FX/FX2 switches; and Cisco Nexus 9500 series switches with X9700-EX and X9700-FX line cards.

Beginning Cisco NX-OS Release 9.3(5), SVI and subinterface unicast counters are supported on Cisco Nexus N9K-C9316D-GX, N9K-C93600CD-GX, N9K-C9364C-GX switches.

Note

- Enabling this feature disables VXLAN, MPLS, Tunnel, Multicast, and ERSPAN counters. Reload the switch for the changes to take effect.
- For a vPC setup, the **peer-gateway** feature must be enabled under the **vpc domain** on both vPC peers. Otherwise, SVI counters may be inconsistent.

To configure SVI and subinterface ingress and/or egress unicast counters on a device, follow these steps:

SUMMARY STEPS

1. **configure terminal**
2. **[no] hardware profile svi-and-si flex-stats-enable**

3. **copy running-config startup-config**
4. **reload**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**[no] hardware profile svi-and-si flex-stats-enable**
	Example:
	`switch(config)# hardware profile svi-and-si flex-stats-enable`
	`switch(config-if)#`
		Configures the ingress/egress unicast counters on SVI and subinterface.
		Note
		You must save the configuration and reload the switch for this command to work.

Step 3	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration.
Step 4	**reload** **Example:** `switch(config-if)# reload`	Reload the switch.

Configuring Subinterface Multicast and Broadcast Counters

Beginning Cisco NX-OS Release 9.3(6), subinterface multicast and broadcast counters are supported on Cisco Nexus N9K-C9336C-FX2 and N9K-C93240YC-FX2 switches.

To configure multicast and broadcast counters on a device, follow these steps:

SUMMARY STEPS

1. **configure terminal**
2. **[no] hardware profile sub-interface flex-stats**
3. **copy running-config startup-config**
4. **reload**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**[no] hardware profile sub-interface flex-stats**
	Example:
	`switch(config)# hardware profile sub-interface flex-stats`
	`switch(config-if)#`
		Enables subinterface flex stats for multicast and broadcast counters.

Step 3	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration.
Step 4	**reload** **Example:** `switch(config-if)# reload`	Reload the switch.

Example

The following example displays the subinterface multicast and broadcast counters as a result of show interface counters command:

```
switch(config)# show int ethernet 1/31/4.1 counters
--------------------------------------------------------------------------------
Port                                    InOctets                    InUcastPkts
--------------------------------------------------------------------------------
Eth1/31/4.1                                    0                              0

--------------------------------------------------------------------------------
Port                                  InMcastPkts                   InBcastPkts
--------------------------------------------------------------------------------
Eth1/31/4.1                                    0                              0

--------------------------------------------------------------------------------
Port                                InIPv4Octets                InIPv4UcastPkts
--------------------------------------------------------------------------------
Eth1/31/4.1                                    0                              0

--------------------------------------------------------------------------------
Port                              InIPv4McastPkts               InIPv4BcastPkts
--------------------------------------------------------------------------------
Eth1/31/4.1                                    0                              0

--------------------------------------------------------------------------------
Port                                InIPv6Octets                InIPv6UcastPkts
```

```
--------------------------------------------------------------------------------
Eth1/31/4.1                                   0                               0

--------------------------------------------------------------------------------
Port                              InIPv6McastPkts                 InIPv6BcastPkts
--------------------------------------------------------------------------------
Eth1/31/4.1                                   0                               0

--------------------------------------------------------------------------------
Port                                   OutOctets                    OutUcastPkts
--------------------------------------------------------------------------------
Eth1/31/4.1                                   0                               0

--------------------------------------------------------------------------------
Port                                 OutMcastPkts                    OutBcastPkts
--------------------------------------------------------------------------------
Eth1/31/4.1                                   0                               0

--------------------------------------------------------------------------------
Port                                 OutIPv4Octets                OutIPv4UcastPkts
--------------------------------------------------------------------------------
Eth1/31/4.1                                   0                               0

--------------------------------------------------------------------------------
Port                               OutIPv4McastPkts                OutIPv4BcastPkts
--------------------------------------------------------------------------------
Eth1/31/4.1                                   0                               0

--------------------------------------------------------------------------------
Port                                 OutIPv6Octets                OutIPv6UcastPkts
--------------------------------------------------------------------------------
Eth1/31/4.1                                   0                               0

--------------------------------------------------------------------------------
Port                               OutIPv6McastPkts                OutIPv6BcastPkts
--------------------------------------------------------------------------------
Eth1/31/4.1                                   0                               0
```

Verifying the Layer 3 Interfaces Configuration

To display the Layer 3 configuration, perform one of the following tasks:

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.

Command	Purpose
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* **status**	Displays the VLAN interface administrative status and protocol status.

Monitoring the Layer 3 Interfaces

Use the following commands to display Layer 3 statistics:

Command	Purpose
load- interval {**interval** *seconds* {**1** | **2** | **3**}}	Cisco Nexus 9000 Series devices set three different sampling intervals to bit-rate and packet-rate statistics. The range for VLAN network interface is 60 to 300 seconds, and the range for Layer interfaces is 30 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**	Displays the Layer 3 interface input and output counters.

Command	Purpose
show interface ethernet errors *slot/port* **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 errors *slot/port* **counters errors**	Displays the Layer 3 interface input and output errors.
show interface ethernet errors *slot/port* **counters snmp**	Displays the Layer 3 interface counters reported by SNMP MIBs.
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* **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 *number* **counters snmp**	Displays the VLAN interface counters reported by SNMP MIBs.

Configuration Examples for Layer 3 Interfaces

This example shows how to configure Ethernet subinterfaces:

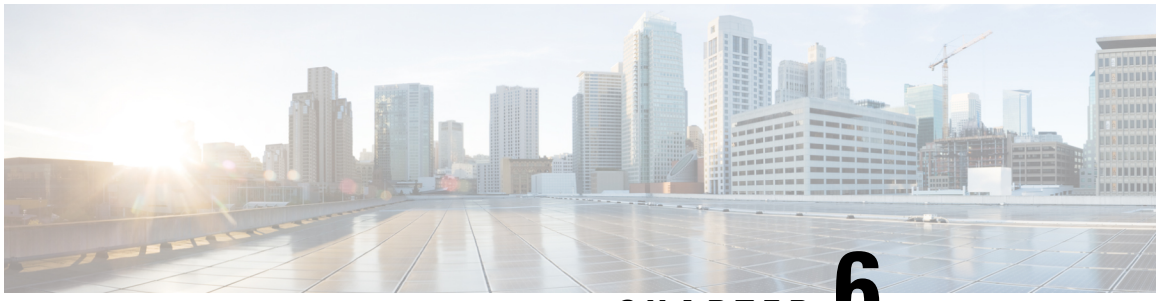
```
interface ethernet 2/1.10
description Layer 3
ip address 192.0.2.1/8
```

This example shows how to configure a loopback interface:

```
interface loopback 3
ip address 192.0.2.2/32
```

Related Documents

Related Documents	Document Title
IP	*Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide*
VLANs	*Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide*



CHAPTER 6

Configuring Bidirectional Forwarding Detection

Bidirectional Forwarding Detection

Bidirectional Forwarding Detection (BFD) is a protocol designed to quickly identify faults in the forwarding path between two devices. BFD simplifies network profiling and planning by offering predictable reconvergence time.

BFD detects forwarding path failures across various media types, encapsulations, topologies, and routing protocols. It provides subsecond failure detection between two adjacent devices, distributing some load onto the data plane on supported modules. BFD can be less CPU-intensive than protocol hello messages.

Asynchronous mode

BFD asynchronous mode is a BFD session mode that:

- involves the exchange of periodic control packets to monitor connectivity,
- establishes and maintains BFD neighbor sessions, and
- negotiates session parameters.

BFD session parameters

The table lists the BFD session parameters and the intervals.

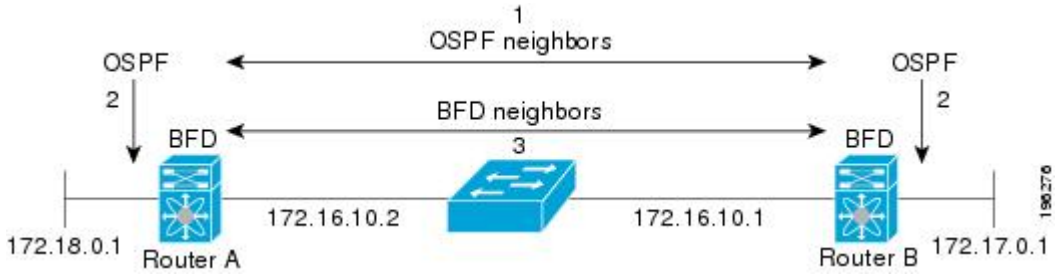
Table 11: BFD session parameters

Session Parameters	Description
Desired minimum transmit interval	The interval at which the device is configured to send BFD hello messages.
Required minimum receive interval	The minimum interval at which the device can accept BFD hello messages from another BFD device.
Detect multiplier	The number of missing BFD hello messages required to detects a fault in the forwarding path.

BFD neighbor workflow

The figure details the BFD neighbor sessions establishment between two routers.

Figure 6: Establishing a BFD Neighbor Relationship



The stages that establish a BFD neighbor session are:

1. The OSPF process discovers a BFD neighbor.
2. The local BFD process gets a request to start a session BFD neighbor session with the OSPF neighbor router.
3. The session is established between the BFD neighbor with the OSPF neighbor router.

BFD Detection of Failures

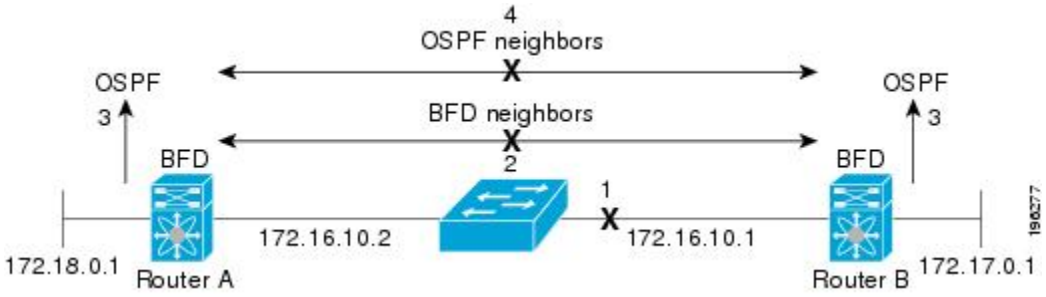
Once a BFD session has been established and timer negotiations are complete, BFD neighbors send BFD control packets that act in the same manner as an IGP hello protocol to detect liveliness, except at a more accelerated rate. BFD detects a failure, but the protocol must take action to bypass a failed peer.

BFD sends a failure detection notice to the BFD-enabled protocols when it detects a failure in the forwarding path. The local device can then initiate the protocol recalculation process and reduce the overall network convergence time.

The following figure shows what happens when a failure occurs in the network (1). The BFD neighbor session with the OSPF neighbor router is torn down (2). BFD notifies the local OSPF process that the BFD neighbor is no longer reachable (3). The local OSPF process tears down the OSPF neighbor relationship (4). If an alternative path is available, the routers immediately start converging on it.

Note Note The BFD failure detection occurs in less than a second, which is much faster than OSPF Hello messages could detect the same failure.

Figure 7: Tearing Down an OSPF Neighbor Relationship



Distributed Operation

Cisco NX-OS can distribute the BFD operation to compatible modules that support BFD. This process offloads the CPU load for BFD packet processing to the individual modules that connect to the BFD neighbors. All BFD session traffic occurs on the module CPU. The module informs the supervisor when a BFD failure is detected.

BFD Echo Function

Echo packets are defined and processed only by the transmitting system. For IPv4 and IPv6, the echo packets' destination address is that of the transmitting device. It is chosen in such a way as to cause the remote system to forward the packet back to the local system. This bypasses the routing lookup on the remote system and relies on the forwarding information base (FIB) instead. BFD can use the slow timer to slow down the asynchronous session when the echo function is enabled and reduce the number of BFD control packets that are sent between two BFD neighbors. The Echo function tests only the forwarding path of the remote system by having the remote (neighbor) system loop them back, so there is less inter-packet delay variability and faster failure detection times.

Security

Cisco NX-OS uses the packet Time to Live (TTL) value to verify that the BFD packets came from an adjacent BFD peer. For all asynchronous and echo request packets, the BFD neighbor sets the TTL value to 255 and the local BFD process verifies the TTL value as 255 before processing the incoming packet. For the echo response packet, BFD sets the TTL value to 254.

You can configure SHA-1 authentication of BFD packets.

High Availability

BFD supports stateless restarts. After a reboot or supervisor switchover, Cisco NX-OS applies the running configuration and BFD immediately sends control packets to the BFD peers.

Virtualization Support

BFD supports virtual routing and forwarding instances (VRFs). VRFs exist within virtual device contexts (VDCs). By default, Cisco NX-OS places you in the default VDC and default VRF.

Prerequisites for BFD

Ensure you meet these prerequisites before you configure BFD.

- Enable the BFD feature.
- Disable ICMP redirect messages on interfaces where BFD is enabled.
- Disable the IP packet verification check for identical IP source and destination addresses.
- Review the detailed prerequisites in the configuration tasks.

Guidelines and Limitations

BFD has the following configuration guidelines and limitations:

- The QSFP 40/100-G BiDi comes up in the highest possible speed available on the port. For example, in the Cisco Nexus 93180LC-EX switch it comes up as 40 G in the first 28 ports and 100 G in the last 4 ports. If you need to connect to 40-G SR4 BiDi, the speed on the 40/100-G BiDi needs to be set to 40 G.
- BFD over private-vlan is not supported Cisco Nexus 9000 Switches.
- Beginning with Cisco NX-OS Release 10.2(1q)F, Layer 3 Unicast BFD is supported on Cisco Nexus N9K-C9332D-GX2B platform switches.
- Forming BFD neighbors on a vPC VLAN through an orphan port is not supported on Cisco Nexus 9000 Switches.
- Beginning with Cisco NX-OS Release 9.2(1), QSFP-40/100-SRBD comes up in the speed of 100-G and inter-operate with other QSFP-40/100-SRBD at either 40-G or 100-G speed on Cisco Nexus 9500 Switches with the N9K-X9636C-RX line card. The QSFP-40/100-SRBD can also inter-operate with QSFP-40G-SR-BD at 40G speeds. However to operate at 40G speed, you must configure the speed as 40G.
- **show** commands with the **internal** keyword are not supported.
- BFD per-member link support is added on Cisco Nexus 9000 Series switches.
- BFD supports BFD version 1.
- BFD supports IPv4 and IPv6.
- BFD supports OSPFv3.
- BFD supports IS-ISv6.
- When configuring BFD over IP unnumbered interfaces, use these guidelines:

- Disable the BFD echo function to prevent the interface from flapping.
- Enable BFD multihop when configuring BGP over IP unnumbered interface.

- Set the **ipv6 nd ns-interval** command range to 15 under the Layer 3 interface configuration to prevent BFD sessions from flapping, when there are a large number of IPv6 adjacencies.

  Alternatively, increase the BFD echo interval to avoid session instability that might occur due to CoPP drops of NS/NA packets.
- BFD supports BGPv6.
- BFD supports EIGRPv6.
- BFD supports only one session per address family, per Layer 3 interface.
- BFD supports only sessions which have unique (src_ip, dst_ip, interface/vrf) combination.
- BFD supports single-hop BFD.
  - Only single-hop static BFD is supported.
  - BFD for BGP supports single-hop EBGP and iBGP peers.
- BFD supports keyed SHA-1 authentication.
- BFD supports the following Layer 3 interfaces—physical interfaces, port channels, sub-interfaces, and VLAN interfaces.
- BFD depends on a Layer 3 adjacency information to discover topology changes, including Layer 2 topology changes. A BFD session on a VLAN interface (SVI) may not be up after the convergence of the Layer 2 topology if there is no Layer 3 adjacency information available.
- For BFD on a static route between two devices, both devices must support BFD. If one or both of the devices do not support BFD, the static routes are not programmed in the Routing Information Base (RIB).
- Both single-hop and multi-hop BFD features are supported with specific restrictions. For multi-hop BFD features restrictions, refer to Guidelines and Limitations for BFD Multihop, on page 168 section.
- Port channel configuration limitations:
  - For Layer 3 port channels used by BFD, you must enable LACP on the port channel.
  - For Layer 2 port channels used by SVI sessions, you must enable LACP on the port channel.
- SVI limitations:
  - An ASIC reset causes traffic disruption for other ports and it can cause the SVI sessions on the other ports to flap. For example, if the carrier interface is a virtual port channel (vPC), BFD is not supported over the SVI interface and it could cause a trigger for an ASIC reset. 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.

    An SVI on the Cisco Nexus series switches should not be configured to establish a BFD neighbor adjacency with a device connected to it via a vPC. This is because the BFD keepalives from the neighbor, if sent over the vPC member link connected to the vPC peer-switch, do not reach this SVI causing the BFD adjacency to fail.

- When you change the topology (for example, add or delete a link into a VLAN, delete a member from a Layer 2 port channel, and so on), the SVI session could be affected. It may go down first and then come up after the topology discovery is finished.
- BFD over FEX HIF interfaces is not supported.
- When a BFD session is over SVI using virtual port-channel (vPC) Peer-Link (either BCM or GEM based ports), the BFD echo function is not supported. You must disable the BFD echo function for all sessions over SVI between vPC peer nodes using the **no bfd echo** command at the SVI configuration level.

Tip If you do not want the SVI sessions to flap and you need to change the topology, you can disable the BFD feature before making the changes and re-enable BFD after the changes have been made. You can also configure the BFD timer to be a large value (for example, 5 seconds), and change it back to a fast timer after the above events complete.

- When you configure the BFD Echo function on the distributed Layer 3 port channels, reloading a member module flaps the BFD session hosted on that module, which results in a packet loss.

  If you connect the BFD peers directly without a Layer 2 switch in between, you can use the BFD per-link mode as an alternative solution.

Note Using BFD per-link mode and sub-interface optimization simultaneously on a Layer 3 port channel is not supported.

- When you specify a BFD neighbor prefix in the **clear** {**ip** | **ipv6**} **route** *prefix* command, the BFD echo session flaps.
- The **clear** {**ip** | **ipv6**} **route *** command causes BFD echo sessions to flap.
- HSRP for IPv4 is supported with BFD.
- BFD packets generated by the Cisco NX-OS device line cards are sent with COS 6/DSCP CS6. The DSCP/COS values for BFD packets are not user configurable.
- When configuring BFDv6 in no-bfd-echo mode, it is recommended to run with timers of 150 ms with a multiplier of 3.
- BFDv6 is not supported for VRRPv3 and HSRP for v6.
- IPv6 **eigrp bfd** cannot be disabled on an interface.
- IETF BFD is not supported on N9K-X96136YC-R, N9K-X9636C-R, N9K-X9636C-RX and N9K-X9636Q-R line cards.
- Port channel configuration notes:
  - When the BFD per-link mode is configured, the BFD echo function is not supported. You must disable the BFD echo function using the **no bfd echo** command before configuring the **bfd per-link** command.

- Before configuring BFD per-link, make sure there is no BFD session running on the port-channel. If there is any BFD session running already, remove it and then proceed with bfd per-link configuration.
- Configuring BFD per-link with link-local is not supported.
- The supported platforms include Cisco Nexus 9500 Series switches with N9K-X9636C-R, N9K-X9636Q-R, N9K-X9636C-RX line cards.

- Beginning with Cisco NX-OS Release 9.3(7), BFD is supported on unnumbered interfaces.

Note BFD over unnumbered Switched Virtual Interfaces (SVIs) are not supported.

Downgrade compatibility for BFD on unnumbered interface support cannot be verified using **show incompatibility nxos bootflash:filename** command. The compatibility will be checked during **install all** command.

- When you configure BFD on a numbered interface along with OSPF and when the interface is converted to an unnumbered interface, the OSPF and BFD command remains in the running configuration but the BFD functionality may not work
- The following BFD command configurations are not supported for configuration replace:
  - **port-channel bfd track-member-link**
  - **port-channel bfd destination** *destination-ip-address*
- Cisco Nexus 9800 platform switches have the following limitation for BFD IPv6 sessions:
  - Each ASIC unit in supervisor switch mode of line card supports a maximum of 256 BFD IPv6 sessions. If more BFD IPv6 sessions are required, sessions must be spread across ASIC units or line cards.
- Beginning with Cisco NX-OS Release 10.3(1)F, BFD supports single-hop BFD on routed port, routed-sub interface, and breakout port of Cisco Nexus 9808 platform switches.
- Use the **bfd authentication interop** command to configure BFD authentication interoperability between Nexus and non-Nexus platforms. If you do not configure this command, BFD authentication fails due to an invalid authentication sequence number field format.
- BFD Authentication is not supported on Cisco Nexus 9800 platform switches.

BFD Support on Nexus Switches

BFD support is available on the Nexus platforms in these releases. For more information, see platform support matrix.

Table 12: BFD Support on Nexus Switches

Platform	Introduced in Cisco NX-OS Release
Nexus 9808	10.3.1F

Platform	Introduced in Cisco NX-OS Release
N9K-C9348D-GX2A N9K-C9364D-GX2A N9K-C9332D-GX2B Cisco Nexus 9300-EX, 9300-FX, 9300-FX2, 9300-FX3, and 9300-GX	10.2.3F
9364C-GX 9316D-GX 93600CD-GX N9K-X96136YC-R, N9K-X9636C-R, N9K-X9636C-RX and N9K-X9636Q-R	9.3.3F

Default Settings

The following table lists the default settings for BFD parameters.

Table 13: Default BFD Parameters

Parameters	Default
BFD feature	Disabled
Required minimum receive interval	50 milliseconds
Desired minimum transmit interval	50 milliseconds
Detect multiplier	3
Echo function	Enabled
Mode	Asynchronous
Port-channel	Logical mode (one session per source-destination pair address)
Slow timer	2000 milliseconds

Configuring BFD

Best Practices for BFD configuration hierarchy and inheritance

Consider these points when you configure BFD at:

- Interface-level configuration versus global configuration

- Member ports and port channels

Interface-level configuration versus global configuration

Configure BFD at both the global level and at the interface level.

Note Interface-level configuration overrides the global configuration.

Inheritance for member ports and port channels

Configure the member port to inherit the BFD configuration of the primary port channel.

Task Flow for Configuring BFD

Follow these steps in the following sections to configure BFD:

- Enabling the BFD Feature.
- Configuring Global BFD Parameters or Configuring BFD on an Interface.

Enable BFD feature

Enable the BFD feature to configure BFD on an interface and protocol.

Procedure

Step 1 Enter the configuration mode with the **configure terminal** command.

Example:

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

Step 2 Enable BFD with the **feature bfd** command.

Example:

```
switch(config)# feature bfd
```

Step 3 (Optional) View the status of features with the **show feature | include bfd** command.

Example:

```
switch(config)# show feature | include
bfd
```

Step 4 (Optional) Save the configuration with the **copy running-config startup-config** command.

Example:

```
switch(config)# copy running-config startup-config
```

Disable BFD

Procedure

	Command or Action	Purpose
Step 1	Disable the BFD feature and remove all associated configurations with the **no feature bfd** command. **Example:** `switch(config)# no feature bfd`	

Configure global BFD parameters

Configure default session behaviors for all BFD (Bidirectional Forwarding Detection) sessions on your device.

BFD global parameters set the timer and detection characteristics for all BFD sessions. You can override these parameters at the interface.

You can configure these settings for all BFD sessions on the device. Both BFD peers negotiate the session parameters in a three-way handshake.

To override these global session parameters on an interface, see Configuring BFD on an Interface.

Use these steps to configure global BFD parameters.

Before you begin

Enable the BFD feature, see Configure global BFD parameters, on page 138

Procedure

Step 1 Enter configuration mode using the **configure terminal** command.

Example:

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

Step 2 Configure the BFD session parameters for all BFD sessions using the **bfd interval** *mintx* **min_rx** *msec* **multiplier** *value* command.

Example:

```
switch(config)# bfd interval 50 min_rx 50 multiplier 3
```

This command overrides the values you configure for BFD session parameters on individual interfaces.

The intervals *mintx* and *msec* range from 50 milliseconds to 999 milliseconds, with a default of 50 milliseconds.

The multiplier ranges from 1 to 50. The default is 3.

Step 3 Configure the slow timer used in the echo function using the **bfd slow-timer** [*interval*] command.

Example:

```
switch(config)# bfd slow-timer 2000
```

This value determines how quickly BFD starts a new session. It specifies the rate at which asynchronous sessions send BFD control packets when the echo function is enabled.

The **slow-timer** value sets the interval for control packets. Echo packets use the configured BFD intervals for link failure detection. Control packets at the slower rate maintain the BFD session.

The range is from 1000 to 30,000 milliseconds. The default is 2000.

Step 4 Configure the interface used for Bidirectional Forwarding Detection (BFD) echo frames **bfd echo-interface loopback** *interface number*

Example:

```
switch(config)# bfd echo-interface loopback 1 3
```

This command changes the source address for the echo packets to the one configured on the specified loopback interface. The interface number range is from 0 to 1023.

Step 5 (Optional) Display the BFD running configuration using the **show running-config bfd** command.

Example:

```
switch(config)# show running-config bfd
```

Step 6 (Optional) Save the configuration using the **copy running-config startup-config** command.

Example:

```
switch(config)# copy running-config startup-config
```

Your device uses the specified default BFD parameters for all BFD sessions unless you override them on an interface.

Example

Configure BFD on an Interface

You can configure the BFD session parameters for all BFD sessions on an interface. The BFD session parameters are negotiated between the BFD peers in a three-way handshake.

This configuration overrides the global session parameters for the configured interface.

Before you begin

Ensure that Internet Control Message Protocol (ICMP) redirect messages are disabled on BFD-enabled interfaces. Use the **no ip redirects** command or the **no ipv6 redirects** command on the interface.

Enable the BFD feature. See the Enabling the BFD Feature section.

Procedure

Step 1 **configure terminal**

Example:

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

Enters configuration mode.

Step 2 **interface** *int-if*

Example:

```
switch(config)# interface ethernet 2/1
switch(config-if)#
```

Enters interface configuration mode. Use the ? keyword to display the supported interfaces.

Step 3 **bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*

Example:

```
switch(config-if)# bfd interval 50
min_rx 50 multiplier 3
```

Configures the BFD session parameters for all BFD sessions on the device. This command overrides these values by configuring the BFD session parameters on an interface. The *mintx* and *msec* range is from 50 to 999 milliseconds and the default is 50. The multiplier range is from 1 to 50. The multiplier default is 3.

Beginning with Cisco NX-OS Release 9.3(5), configuring BFD session parameters under interface with default timer values using the **bfd interval 50 min_rx 50 multiplier 3** command is functionally equivalent to **no bfd interval** command.

Once BFD session parameters under interface are set to default values, those BFD sessions running on that interface will inherit global session parameters, if present.

Step 4 **bfd authentication keyed-sha1 keyid** *id* **key** *ascii_key*

Example:

```
switch(config-if)# bfd authentication
keyed-sha1 keyid 1 ascii_key cisco123
```

(Optional) Configures SHA-1 authentication for all BFD sessions on the interface. The *ascii_key* string is a secret key shared among BFD peers. The *id* value, a number between 0 and 255, is assigned to this particular *ascii_key* . BFD packets specify the key by *id* , allowing the use of multiple active keys.

To disable SHA-1 authentication on the interface, use the **no** form of the command.

Step 5 Use the **bfd authentication interop** command to configure BFD authentication interoperability between Nexus and non-Nexus platforms.

Example:

```
switch(config-if)# bfd authentication interop
```

Step 6 **show running-config bfd**

Example:

```
switch(config-if)# show running-config bfd
```

(Optional) Displays the BFD running configuration.

Step 7 **copy running-config startup-config**

Example:

```
switch(config-if)# copy running-config startup-config
```

(Optional) Saves the configuration change.

Example

What to do next

-

Configuring BFD on a Port Channel

You can configure the BFD session parameters for all BFD sessions on a port channel. If per-link mode is used for Layer 3 port channels, BFD creates a session for each link in the port channel and provides an aggregate result to client protocols. For example, if the BFD session for one link on a port channel is up, BFD informs client protocols, such as OSPF, that the port channel is up. The BFD session parameters are negotiated between the BFD peers in a three-way handshake.

This configuration overrides the global session parameters for the configured port channel. The member ports of the port channel inherit the port channel BFD session parameters.

Before you begin

Ensure that you enable LACP on the port channel before you enable BFD.

Ensure that Internet Control Message Protocol (ICMP) redirect messages are disabled on BFD-enabled interfaces. Use the **no ip redirects** command on the interface.

Enable the BFD feature. See the Enabling the BFD Feature section.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *number*
3. **bfd per-link**
4. **bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*
5. **bfd authentication keyed-sha1 keyid** *id* **key** *ascii_key*
6. **show running-config bfd**
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 2`
	`switch(config-if)#`
		Enters port-channel configuration mode. Use the **?** keyword to display the supported number range.

Step 3	**bfd per-link** **Example:** `switch(config-if)# bfd per-link`	Configures the BFD sessions for each link in the port channel.
Step 4	**bfd interval** *mintx* **min_rx** *msec* **multiplier** *value* **Example:** `switch(config-if)# bfd interval 50 min_rx 50 multiplier 3`	(Optional) Configures the BFD session parameters for all BFD sessions on the port channel. This command overrides these values by configuring the BFD session parameters. The *mintx* and *msec* range is from 50 to 999 milliseconds and the default is 50. The multiplier range is from 1 to 50. The multiplier default is 3.
Step 5	**bfd authentication keyed-sha1 keyid** *id* **key** *ascii_key* **Example:** `switch(config-if)# bfd authentication keyed-sha1 keyid 1 ascii_key cisco123`	(Optional) Configures SHA-1 authentication for all BFD sessions on the interface. The *ascii_key* string is a secret key shared among BFD peers. The *id* value, a number between 0 and 255, is assigned to this particular *ascii_key*. BFD packets specify the key by *id*, allowing the use of multiple active keys. To disable SHA-1 authentication on the interface, use the **no** form of the command.
Step 6	**show running-config bfd** **Example:** `switch(config-if)# show running-config bfd`	(Optional) Displays the BFD running configuration.
Step 7	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

Configure the BFD Echo function (task)

You can configure the BFD echo function on one or both ends of a BFD-monitored link. The echo function slows down the required minimum receive interval, based on the configured slow timer. The RequiredMinEchoRx BFD session parameter remains nonzero if you disable the echo function to comply with RFC 5880 When you enable the echo function, the slow timer value becomes the required minimum receive interval.

Before you begin

Enable the BFD feature. See the Enable BFD feature.

Configure the BFD session parameters. See Configuring Global BFD Parameters or Configuring BFD on an Interface.

Disable Internet Control Message Protocol (ICMP) redirect messages on BFD-enabled interfaces using the **no ip redirects** command on the interface.

Procedure

Step 1 Enter the configuration mode using the **configure terminal** command.

Example:

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

Step 2 Set the slow timer to determine when BFD starts a new sesion using the **bfd slow-timer** *echo-interval* command.

Example:

```
switch(config)# bfd slow-timer 2000
```

When the BFD echo function is enabled, this value also slows down the asynchronous sessions.

This value overwrites the required minimum receive interval when the echo function is enabled. The range is from 1000 to 30,000 milliseconds. The default is 2000 milliseconds.

Step 3 Enters interface configuration mode using the **interface** *int-if* command.

Example:

```
switch(config)# interface ethernet 2/1
switch(config-if)#
```

Use the ? keyword to display the supported interfaces.

Step 4 Enable the echo function using the **bfd echo** command.

Example:

```
switch(config-if)# bfd echo
```

The default is enabled.

Step 5 (Optional) Display the BFD running configuration using the **show running-config bfd** command.

Example:

```
switch(config-if)# show running-config bfd
```

Step 6 (Optional) Saves the configuration using the **copy running-config startup-config** command.

Example:

```
switch(config-if)# copy running-config startup-config
```

Configuring Per-Member Link BFD Sessions

BFD per-member link support is added on Cisco Nexus 9000 Series switches. See the following sections for more information.

BFD Enhancement to Address Per-link Efficiency

The Bidirectional Forwarding (BFD) enhancement to address per-link efficiency, called as IETF Micro BFD, lets you configure the individual BFD sessions on every Link Aggregation Group (LAG) member interfaces (as defined in RFC 7130).

With this enhancement, the BFD sessions run on each member link of the port-channel. If BFD detects a link failure, the member link is removed from the forwarding table. This mechanism delivers faster failure detection as the BFD sessions are created on an individual port-channel interface.

The BFD sessions running on member links of the port-channel are called as Micro BFD sessions. You can configure RFC 7130 BFD over main port-channel interface, that performs bandwidth monitoring over LAG by having one Micro BFD session over each member. If any of the member port goes down, the port is removed from the forwarding table and this prevents traffic disruption on that member.

Micro BFD sessions are supported for both LACP and non-LACP based-port channels. For more information on how to configure Micro BFD sessions, see *Configuring Micro BFD Sessions*.

Limitations of the IETF Bidirectional Forwarding Detection

See the following limitations of the IETF Bidirectional Forwarding Detection:

- BFD Limitations
  - IETF Micro-BFD sessions supports only single-hop BFD sessions. We recommend that you do *not* configure IPs from different subnets to establish the Micro-BFD sessions.
  - It cannot co-exist with BFD over logical port-channels or proprietary BFD per-member links. BFD IPv6 logical/proprietary per-link session is also not supported when BFD IETF IPv4 is configured on PC.
  - When you configure logical BFD session under any routing protocol, make sure that is not applied to any IETF port-channel. Having both logical and IETF configuration for same port-channel results in undefined behavior during ISSU/reloads.
  - IETF BFD IPv6 is not supported.
  - Echo functionality is not supported for Micro-BFD sessions.
  - Port-channel interfaces should be directly connected between two switches that are running the BFD sessions. No intermediate Layer 2 switches are expected.
- EthPCM/LACP Limitations

- If a LACP port-channel has members in hot-standby state, BFD failure in one of the active links may not cause the hot-standby link to come up directly. Once the active link with BFD failure goes down, the hot-standby member becomes active. However, it may not be able to prevent the port-channel from going down before the hot-standby link comes up, in cases where port-channel min-link condition is hit.

- General Limitations:
  - It is supported only on Layer 3 port-channels.
  - It is not supported on the following:
    - vPC
    - Layer 3 sub-interfaces
    - Layer 2 port-channels/Layer 2 Fabric Path
    - FPC/HIF PC
    - Layer 3 sub-interfaces
    - SVI over port-channels

Guidelines for Migration/Configuration of IETF Per-Member Sessions:

See the following guidelines for migration/configuration of IETF per-member sessions:

- The logical BFD sessions that are created using the routing protocols over port-channel sub-interfaces (where RFC 7130 cannot run) are still supported. The main port-channel interface however does not support both logical and RFC 7130 sessions that co-exist. It can support only either of them.
- You can configure RFC 7130 BFD over the main port-channel interface that perform bandwidth monitoring over the LAG by having one Micro-BFD session over each member. If any of the member port goes down, BFD notifies it to the port-channel manager that removes the port from the LTL, thereby preventing blackholing of the traffic on that member.
- If the minimum number of links required to have the port-channel operationally *up* is not met in the above case, the port-channel is brought down by the port-channel manager. This in turn brings down the port-channel sub-interfaces if they are configured and thereby the logical BFD session also comes down notifying the routing protocol.
- When you are using RFC 7130 on the main port-channel and logical BFD on the sub-interfaces, the logical BFD session should be run with lesser aggressive timers than the RFC 7130 BFD session. You can have RFC 7130 configured on the port-channel interface or you can have it configured in conjunction with the logical BFD sessions on the port-channel sub-interfaces.
- When a proprietary per-link is configured, enabling IETF Micro-BFD sessions is not allowed on a port channel and vice-versa. You have to remove the proprietary per-link configuration. Current implementation of proprietary per-link does not allow changing the configuration (no per-link), if there is any BFD session that is bootstrapped by the applications. You need to remove the BFD tracking on the respective applications and remove per-link configuration. The migration path from the proprietary per-link to IETF Micro-BFD is as follows:
  - Remove the BFD configuration on the applications.

- Remove the per-link configuration.
- Enable the IETF Micro-BFD command.
- Enable BFD on the applications.

The same migration path can be followed for proprietary BFD to IETF Micro-BFD on the main port-channel interface.

Configuring Port Channel Interface

Before you begin

Ensure that the BFD feature is enabled.

SUMMARY STEPS

1. switch(config)# **interface port-channel** *port-number*
2. switch(config-if)# **no switchport**

DETAILED STEPS

Procedure

Step 1 switch(config)# **interface port-channel** *port-number*

Configures interface port-channel.

Step 2 switch(config-if)# **no switchport**

Configures interface as Layer 3 port-channel.

What to do next

- Configuring BFD Start Timer
- Enabling IETF Per-link BFD

(Optional) Configuring BFD Start Timer

Complete the following steps to configure the BFD start timer:

SUMMARY STEPS

1. switch(config-if)# **port-channel bfd start** *60*

DETAILED STEPS

Procedure

switch(config-if)# **port-channel bfd start** *60*

Configures the BFD start timer for a port-channel.

Note
The default value is infinite (that is no timer is running). The range of BFD Start Timer value for port-channel is from 60 to 3600 seconds. For start timer to work, configure start timer value before completing the port-channel BFD configurations (that is before port-channel bfd track-member-link and port-channel bfd destination are configured for Layer 3 port-channel interface with the active members).

What to do next

- Enabling IETF Per-link BFD
- Configuring BFD Destination IP Address

Enabling IETF Per-link BFD

SUMMARY STEPS

1. switch(config-if)# **port-channel bfd track-member-link**

DETAILED STEPS

Procedure

switch(config-if)# **port-channel bfd track-member-link**

Enables IETF BFD on port-channel interface.

What to do next

- Configuring BFD Destination IP Address
- Verifying Micro BFD Session Configurations

Configuring BFD Destination IP Address

Complete the following steps to configure the BFD destination IP address:

SUMMARY STEPS

1. switch(config-if)# **port-channel bfd destination***ip-address*

DETAILED STEPS

Procedure

switch(config-if)# **port-channel bfd destination***ip-address*

Configures an IPv4 address to be used for the BFD sessions on the member links.

What to do next

- Verifying Micro BFD Sessions Configuration

Verifying Micro BFD Session Configurations

Use the following commands to verify the Micro BFD session configurations.

SUMMARY STEPS

1. Displays the port-channel and port-channel member operational state.
2. switch# **show bfd neighbors**
3. switch# **show bfd neighbors details**
4. switch# **show tech-support bfd**
5. switch# **show tech-support lacp all**
6. switch# **show running-config interface port-channel** *port-channel-number*

DETAILED STEPS

Procedure

Step 1 Displays the port-channel and port-channel member operational state.

switch# **show port-channel summary**

Step 2 switch# **show bfd neighbors**

Displays Micro BFD sessions on port-channel members.

Step 3 switch# **show bfd neighbors details**

Displays BFD session for a port channel interface and the associated Micro BFD sessions on members.

Step 4 switch# **show tech-support bfd**

Displays the technical support information for BFD.

Step 5 switch# **show tech-support lacp all**

Displays the technical support information for Ethernet Port Manager, Ethernet Port-channel Manager, and LACP.

Step 6 switch# **show running-config interface port-channel** *port-channel-number*

Displays the running configuration information of the port-channel interface.

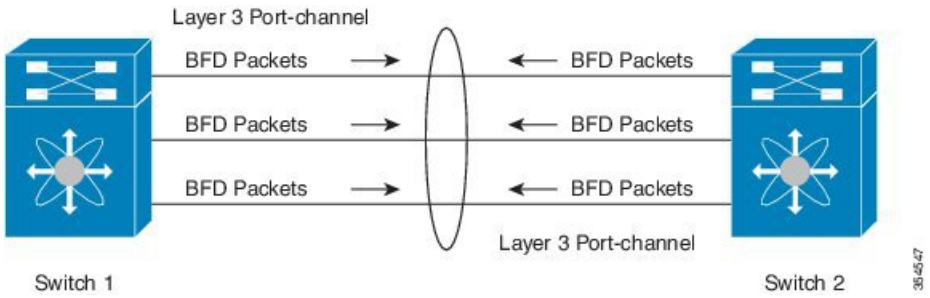
Examples: Configuring Micro BFD Sessions

See the following examples for configuring Micro BFD sessions.

Configuring Micro BFD Sessions

In this example, the following topology is used.

Figure 8: Configuring Micro BFD Session



The sample configuration of switch 1 is as follows:

```
feature bfd
configure terminal
    interface port-channel 10
       port-channel bfd track-member-link
       port-channel bfd destination 10.1.1.2
       port-channel bfd start 60
       ip address 10.1.1.1/24
```

The sample configuration of switch 2 is as follows:

```
feature bfd
configure terminal
     interface port-channel 10
           port-channel bfd track-member-link
           port-channel bfd destination 10.1.1.1
           port-channel bfd start 60
             ip address 10.1.1.2/24
```

Verifying Micro BFD Sessions Configuration

The following example displays the show output of the **show running-config interface port-channel**<*port-channel*>, **show port-channel summary**, **show bfd neighbors vrf**

internet_routes, and **show bfd neighbors interface port-channel** *<port-channel>* **vrf internet_routes details** commands.

```
switch# show running-config interface port-channel 1001

!Command: show running-config interface port-channel1001
!Time: Fri Oct 21 09:08:00 2016

version 7.0(3)I5(1)

interface port-channel1001
  no switchport
  vrf member internet_routes
  port-channel bfd track-member-link
  port-channel bfd destination 40.4.1.2
  ip address 40.4.1.1/24
  ipv6 address 2001:40:4:1::1/64

switch# show por
port-channel   port-profile
switch# show port-channel summary
Flags:  D - Down        P - Up in port-channel (members)
        I - Individual  H - Hot-standby (LACP only)
        s - Suspended   r - Module-removed
        b - BFD Session Wait
        S - Switched    R - Routed
        U - Up (port-channel)
        p - Up in delay-lacp mode (member)
        M - Not in use. Min-links not met
--------------------------------------------------------------------------------
Group Port-       Type     Protocol  Member Ports
      Channel
--------------------------------------------------------------------------------
1001  Po1001(RU)  Eth      LACP      Eth1/11/1(P)  Eth1/11/2(P)  Eth1/12/1(P)
                                     Eth1/12/2(P)
switch# show bfd neighbors vrf internet_routes

OurAddr         NeighAddr       LD/RD                RH/RS           Holdown(mult)
State       Int                   Vrf
40.4.1.1        40.4.1.2        1090519041/0         Up              N/A(3)            Up
          Po1001                internet_routes
40.4.1.1        40.4.1.2        1090519042/1090519051 Up             819(3)            Up
          Eth1/12/1             internet_routes
40.4.1.1        40.4.1.2        1090519043/1090519052 Up             819(3)            Up
          Eth1/12/2             internet_routes
40.4.1.1        40.4.1.2        1090519044/1090519053 Up             819(3)            Up
          Eth1/11/1             internet_routes
40.4.1.1        40.4.1.2        1090519045/1090519054 Up             819(3)            Up
          Eth1/11/2             internet_routes
switch#

switch# show bfd neighbors interface port-channel 1001 vrf internet_routes details

OurAddr         NeighAddr       LD/RD                RH/RS           Holdown(mult)
State       Int                   Vrf
40.4.1.1        40.4.1.2        1090519041/0         Up              N/A(3)            Up
          Po1001                internet_routes

Session state is Up
Local Diag: 0
Registered protocols:  eth_port_channel
Uptime: 1 days 11 hrs 4 mins 8 secs
Hosting LC: 0, Down reason: None, Reason not-hosted: None
Parent session, please check port channel config for member info
```

```
switch#

switch# show bfd neighbors interface ethernet 1/12/1 vrf internet_routes details

OurAddr         NeighAddr       LD/RD                 RH/RS           Holdown(mult)
State       Int                   Vrf
40.4.1.1        40.4.1.2        1090519042/1090519051 Up              604(3)            Up
          Eth1/12/1             internet_routes

Session state is Up and not using echo function
Local Diag: 0, Demand mode: 0, Poll bit: 0, Authentication: None
MinTxInt: 100000 us, MinRxInt: 100000 us, Multiplier: 3
Received MinRxInt: 300000 us, Received Multiplier: 3
Holdown (hits): 900 ms (0), Hello (hits): 300 ms (458317)
Rx Count: 427188, Rx Interval (ms) min/max/avg: 19/1801/295 last: 295 ms ago
Tx Count: 458317, Tx Interval (ms) min/max/avg: 275/275/275 last: 64 ms ago
Registered protocols:  eth_port_channel
Uptime: 1 days 11 hrs 4 mins 24 secs
Last packet: Version: 1               - Diagnostic: 0
             State bit: Up            - Demand bit: 0
             Poll bit: 0              - Final bit: 0
             Multiplier: 3            - Length: 24
             My Discr.: 1090519051    - Your Discr.: 1090519042
             Min tx interval: 300000  - Min rx interval: 300000
             Min Echo interval: 300000 - Authentication bit: 0
Hosting LC: 1, Down reason: None, Reason not-hosted: None
Member session under parent interface Po1001


switch# show bfd neighbors interface ethernet 1/12/2 vrf internet_routes details

OurAddr         NeighAddr       LD/RD                 RH/RS           Holdown(mult)
State       Int                   Vrf
40.4.1.1        40.4.1.2        1090519043/1090519052 Up              799(3)            Up
          Eth1/12/2             internet_routes

Session state is Up and not using echo function
Local Diag: 0, Demand mode: 0, Poll bit: 0, Authentication: None
MinTxInt: 100000 us, MinRxInt: 100000 us, Multiplier: 3
Received MinRxInt: 300000 us, Received Multiplier: 3
Holdown (hits): 900 ms (0), Hello (hits): 300 ms (458336)
Rx Count: 427207, Rx Interval (ms) min/max/avg: 19/1668/295 last: 100 ms ago
Tx Count: 458336, Tx Interval (ms) min/max/avg: 275/275/275 last: 251 ms ago
Registered protocols:  eth_port_channel
Uptime: 1 days 11 hrs 4 mins 30 secs
Last packet: Version: 1               - Diagnostic: 0
             State bit: Up            - Demand bit: 0
             Poll bit: 0              - Final bit: 0
             Multiplier: 3            - Length: 24
             My Discr.: 1090519052    - Your Discr.: 1090519043
             Min tx interval: 300000  - Min rx interval: 300000
             Min Echo interval: 300000 - Authentication bit: 0
Hosting LC: 1, Down reason: None, Reason not-hosted: None
Member session under parent interface Po1001
switch#
```

Configuring BFD Support for Routing Protocols

Configuring BFD on BGP

You can configure BFD for the Border Gateway Protocol (BGP).

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Enable the BGP feature. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *as-number*
3. **neighbor** (*ip-address* | *ipv6-address*) **remote-as** *as-number*
4. **bfd** [**multihop** | **singlehop**]
5. **update-source** *interface*
6. **show running-config bgp**
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**router bgp** *as-number*
	Example:
	`switch(config)# router bgp 64496`
	`switch(config-router)#`
		Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 3	**neighbor** (*ip-address* | *ipv6-address*) **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 209.165.201.1 remote-as 64497`
	`switch(config-router-neighbor)#`
		Configures the IPv4 or IPv6 address and AS number for a remote BGP peer. The *ip-address* format is x.x.x.x. The *ipv6-address* format is A:B::C:D.

	Command or Action	Purpose
Step 4	**bfd** [**multihop** | **singlehop**] **Example:** `switch(config-router-neighbor)# bfd multiihop`	Configures the BFD multi hop or single hop session on the device. The default is with no keyword. When you do not specify any keyword and if the peer is directly connected then a single hop session is selected, if the peer is not connected then a multi hop session type is selected. When you specify a "multihop" or "singlehop" option, the session type is forced in a device according to the CLI option.
Step 5	**update-source** *interface* **Example:** `switch(config-router-neighbor)# update-source ethernet 2/1`	Allows BGP sessions to use the primary IP address from a particular interface as the local address when forming a BGP session with a neighbor and enables BGP to register as a client with BFD.
Step 6	**show running-config bgp** **Example:** `switch(config-router-neighbor)# show running-config bgp`	(Optional) Displays the BGP running configuration.
Step 7	**copy running-config startup-config** **Example:** `switch(config-router-neighbor)# copy running-config startup-config`	(Optional) Saves the configuration change.

Configuring BFD on EIGRP

You can configure BFD for the Enhanced Interior Gateway Routing Protocol (EIGRP).

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Enable the EIGRP feature. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **router eigrp** *instance-tag*
3. **bfd** [**ipv4** | **ipv6**]
4. **interface** *int-if*
5. **ip eigrp** *instance-tag* **bfd**
6. **show ip eigrp** [**vrf** *vrf-name*] [**interfaces** *if*]
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**router eigrp** *instance-tag*
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Creates a new EIGRP process with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
		If you configure an instance-tag that does not qualify as an AS number, you must use the **autonomous-system** to configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.

Step 3	**bfd** [**ipv4** | **ipv6**] **Example:** `switch(config-router-neighbor)# bfd ipv4`	(Optional) Enables BFD for all EIGRP interfaces.

Step 4	**interface** *int-if*
	Example:
	`switch(config-router-neighbor)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.

Step 5	**ip eigrp** *instance-tag* **bfd** **Example:** `switch(config-if)# ip eigrp Test1 bfd`	(Optional) Enables or disables BFD on an EIGRP interface. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters. The default is disabled.
Step 6	**show ip eigrp** [**vrf** *vrf-name*] [**interfaces** *if*] **Example:** `switch(config-if)# show ip eigrp`	(Optional) Displays information about EIGRP. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
Step 7	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

Configuring BFD on OSPF

You can configure BFD for the Open Shortest Path First.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Enable the OSPF feature. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*
3. **bfd** [**ipv4** | **ipv6**]
4. **interface** *int-if*
5. **ip ospf bfd**
6. **show ip ospf** [**vrf** *vrf-name*] [**interfaces** *if*]
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 200`
	`switch(config-router)#`
		Creates a new OSPF instance with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.

Step 3	**bfd** [**ipv4**	**ipv6**] **Example:** `switch(config-router)# bfd`	(Optional) Enables BFD for all OSPF interfaces.

Step 4	**interface** *int-if*
	Example:
	`switch(config-router)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.

Step 5	**ip ospf bfd** **Example:** `switch(config-if)# ip ospf bfd`	(Optional) Enables or disables BFD on an OSPF interface. The default is disabled.

	Command or Action	Purpose
Step 6	**show ip ospf** [**vrf** *vrf-name*] [**interfaces** *if*] **Example:** `switch(config-if)# show ip ospf`	(Optional) Displays information about OSPF. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
Step 7	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

Example Configurations for BFD on OSPF

Example configuration where BFD is enabled under a non-default VRF (OSPFv3 neighbors in vrf3).

```
configure terminal
  router ospfv3 10
    vrf vrf3
    bfd
```

Configuring BFD on IS-IS

You can configure BFD for the Intermediate System-to-Intermediate System (IS-IS) protocol.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Enable the IS-IS feature. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **router isis** *instance-tag*
3. **bfd** [**ipv4** | **ipv6**]
4. **interface** *int-if*
5. **isis bfd**
6. **show isis** [**vrf** *vrf-name*] [**interface** *if*]
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router isis** *instance-tag*
	Example:
	`switch(config)# router isis 100`
	`switch(config-router)# net 49.0001.1720.1600.1001.00`
	`switch(config-router)# address-family ipv6 unicast`
		Creates a new IS-IS instance with the configured *instance tag*.

Step 3	**bfd** [**ipv4** | **ipv6**] **Example:** `switch(config-router)# bfd`	(Optional) Enables BFD for all OSPF interfaces.

Step 4	**interface** *int-if*
	Example:
	`switch(config-router)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.

Step 5	**isis bfd** **Example:** `switch(config-if)# isis bfd`	(Optional) Enables or disables BFD on an IS-IS interface. The default is disabled.
Step 6	**show isis** [**vrf** *vrf-name*] [**interface** *if*] **Example:** `switch(config-if)# show isis`	(Optional) Displays information about IS-IS. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
Step 7	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

Example Configurations for BFD on IS-IS

Example configuration for IS-IS where BFD is enabled under IPv4 and an IPv6 address family.

```
configure terminal
  router isis isis-1
```

```
bfd
address-family ipv6 unicast
bfd
```

Configuring BFD on HSRP

You can configure BFD for the Hot Standby Router Protocol (HSRP). The active and standby HSRP routers track each other through BFD. If BFD on the standby HSRP router detects that the active HSRP router is down, the standby HSRP router treats this event as an active time rexpiry and takes over as the active HSRP router.

The **show hsrp detail** command shows this event as BFD@Act-down or BFD@Sby-down.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Enable the HSRP feature. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **hsrp bfd all-interfaces**
3. **interface** *int-if*
4. **hsrp bfd**
5. **show running-config hsrp**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**hsrp bfd all-interfaces** **Example:** `switch# hsrp bfd all-interfaces`	(Optional) Enables or disables BFD on all HSRP interfaces. The default is disabled.
Step 3	**interface** *int-if* **Example:**	Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.

	Command or Action	Purpose
	`switch(config-router)# interface ethernet 2/1` `switch(config-if)#`	
Step 4	**hsrp bfd** **Example:** `switch(config-if)# hsrp bfd`	(Optional) Enables or disables BFD on an HSRP interface. The default is disabled.
Step 5	**show running-config hsrp** **Example:** `switch(config-if)# show running-config hsrp`	(Optional) Displays the HSRP running configuration.
Step 6	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

Configuring BFD on VRRP

You can configure BFD for the Virtual Router Redundancy Protocol (VRRP). The active and standby VRRP routers track each other through BFD. If BFD on the standby VRRP router detects that the active VRRP router is down, the standby VRRP router treats this event as an active time rexpiry and takes over as the active VRRP router.

The **show vrrp detail** command shows this event as BFD@Act-down or BFD@Sby-down.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Enable the VRRP feature. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *int-if*
3. **vrrp** *group-no*
4. **vrrp bfd** *address*
5. **show running-config vrrp**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *int-if*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.

Step 3	**vrrp** *group-no* **Example:** `switch(config-if)# vrrp 2`	Specifies the VRRP group number.
Step 4	**vrrp bfd** *address* **Example:** `switch(config-if)# vrrp bfd`	Enables or disables BFD on a VRRP interface. The default is disabled.
Step 5	**show running-config vrrp** **Example:** `switch(config-if)# show running-config vrrp`	(Optional) Displays the VRRP running configuration.
Step 6	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves the configuration change.

Configuring BFD on PIM

You can configure BFD for the Protocol Independent Multicast (PIM) protocol.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Enable the PIM feature. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information.

SUMMARY STEPS

1. **configure terminal**
2. **ip pim bfd**

3. **interface** *int-if*
4. **ip pim bfd-instance** [**disable**]
5. **show running-config pim**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**ip pim bfd** **Example:** `switch(config)# ip pim bfd`	Enables BFD for PIM.

Step 3	**interface** *int-if*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.

Step 4	**ip pim bfd-instance** [**disable**] **Example:** `switch(config-if)# ip pim bfd-instance`	(Optional) Enables or disables BFD on a PIM interface. The default is disabled.
Step 5	**show running-config pim** **Example:** `switch(config)# show running-config pim`	(Optional) Displays the PIM running configuration.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves the configuration change.

Configuring BFD on Static Routes

You can configure BFD for static routes on an interface. You can optionally configure BFD on a static route within a virtual routing and forwarding (VRF) instance.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

SUMMARY STEPS

1. **configure terminal**
2. **vrf context** *vrf-name*
3. **ip route** *route interface* {*nh-address* | *nh-prefix*}
4. **ip route static bfd** *interface* {*nh-address* | *nh-prefix*}
5. **show ip route static** [**vrf** *vrf-name*]
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vrf context** *vrf-name*
	Example:
	`switch(config)# vrf context Red`
	`switch(config-vrf)#`
		(Optional) Enters VRF configuration mode.

Step 3	**ip route** *route interface* {*nh-address*	*nh-prefix*} **Example:** `switch(config-vrf)# ip route 192.0.2.1 ethernet 2/1 192.0.2.4`	Creates a static route Use the **?** keyword to display the supported interfaces.
Step 4	**ip route static bfd** *interface* {*nh-address*	*nh-prefix*} **Example:** `switch(config-vrf)# ip route static bfd ethernet 2/1 192.0.2.4`	Enables BFD for all static routes on an interface. Use the**?** keyword to display the supported interfaces.
Step 5	**show ip route static** [**vrf** *vrf-name*] **Example:** `switch(config-vrf)# show ip route static vrf Red`	(Optional) Displays the static routes.
Step 6	**copy running-config startup-config** **Example:** `switch(config-vrf)# copy running-config startup-config`	(Optional) Saves the configuration change.

Disabling BFD on an Interface

You can selectively disable BFD on an interface for a routing protocol that has BFD enabled at the global or VRF level.

To disable BFD on an interface, use one of the following commands in interface configuration mode:

Command	Purpose
ip eigrp *instance-tag* **bfd disable** **Example:** `switch(config-if)# ip eigrp Test1 bfd disable`	Disables BFD on an EIGRP interface. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
ip ospf bfd disable **Example:** `switch(config-if)# ip ospf bfd disable`	Disables BFD on an OSPFv2 interface.
isis bfd disable **Example:** `switch(config-if)# isis bfd disable`	Disables BFD on an IS-IS interface.

Disabling BFD on an Interface

Example configuration where BFD is disabled per interface.

```
configure terminal
  interface port-channel 10
    no ip redirects
    ip address 22.1.10.1/30
    ipv6 address 22:1:10::1/120
    no ipv6 redirects
    ip router ospf 10 area 0.0.0.0
    ip ospf bfd disable        /*** disables IPv4 BFD session for OSPF
    ospfv3 bfd disable         /*** disables IPv6 BFD session for OSPFv3
```

Configuring BFD Interoperability

Configuring BFD Interoperability in Cisco NX-OS Devices in a Point-to-Point Link

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *int-if*
3. **ip ospf bfd**
4. **no ip redirects**

5. **bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*
6. **exit**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *int-if* **Example:** `switch(config-if)# interface ethernet 2/1`	Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.
Step 3	**ip ospf bfd** **Example:** `switch(config-if)# ip ospf bfd`	Enables BFD on an OSPFv2 interface. The default is disabled. OSPF is used as an example. You can enable BFD of any of the supported protocols.
Step 4	**no ip redirects** **Example:** `switch(config-if)# no ip redirects`	Prevents the device from sending redirects.
Step 5	**bfd interval** *mintx* **min_rx** *msec* **multiplier** *value* **Example:** `switch(config-if)# bfd interval 50 min_rx 50 multiplier 3`	Configures the BFD session parameters for all BFD sessions on the port channel. This command overrides these values by configuring the BFD session parameters. The *mintx* and *msec* range is from 50 to 999 milliseconds and the default is 50. The multiplier range is from 1 to 50. The multiplier default is 3.
Step 6	**exit** **Example:** `switch(config-if)# exit`	Exits interface configuration mode and returns to EXEC mode.

Configuring BFD Interoperability in Cisco NX-OS Devices in a Switch Virtual Interface

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *vlan vlan-id*
3. **bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*

4. **no ip redirects**
5. **ip address** *ip-address/length*
6. **ip ospf bfd**
7. **exit**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *vlan vlan-id*
	Example:
	`switch(config)# interface vlan 998`
	`switch(config-if)#`
		Creates a dynamic Switch Virtual Interface (SVI).

Step 3	**bfd interval** *mintx* **min_rx** *msec* **multiplier** *value* **Example:** `switch(config-if)# bfd interval 50 min_rx 50 multiplier 3`	Configures the BFD session parameters for all BFD sessions on the device. The *mintx* and *msec* range is from 50 to 999 milliseconds and the default is 50. The multiplier range is from 1 to 50. The multiplier default is 3.
Step 4	**no ip redirects** **Example:** `switch(config-if)# no ip redirects`	Prevents the device from sending redirects.
Step 5	**ip address** *ip-address/length* **Example:** `switch(config-if)# ip address 10.1.0.253/24`	Configures an IP address for this interface.
Step 6	**ip ospf bfd** **Example:** `switch(config-if)# ip ospf bfd`	Enables BFD on an OSPFv2 interface. The default is disabled.
Step 7	**exit** **Example:** `switch(config-if)# exit`	Exits interface configuration mode and returns to EXEC mode.

Configuring BFD Interoperability in Cisco NX-OS Devices in Logical Mode

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *type number.subinterface-id*
3. **bfd interval** *mintx* **min_rx** *msec* **multiplier** *value*
4. **no ip redirects**
5. **ip ospf bfd**
6. **exit**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *type number.subinterface-id* **Example:** `switch(config-if)# interface port-channel 50.2`	Enters port channel configuration mode. Use the **?** keyword to display the supported number range.
Step 3	**bfd interval** *mintx* **min_rx** *msec* **multiplier** *value* **Example:** `switch(config-if)# bfd interval 50 min_rx 50 multiplier 3`	Configures the BFD session parameters for all BFD sessions on the port channel. The *mintx* and *msec* range is from 50 to 999 milliseconds and the default is 50. The multiplier range is from 1 to 50. The multiplier default is 3.
Step 4	**no ip redirects** **Example:** `switch(config-if)# no ip redirects`	Prevents the device from sending redirects.
Step 5	**ip ospf bfd** **Example:** `switch(config-if)# ip ospf bfd`	Enables BFD on an OSPFv2 interface. The default is disabled. OSPF is used as an example. You can enable BFD of any of the supported protocols.
Step 6	**exit** **Example:** `switch(config-if)# exit`	Exits interface configuration mode and returns to EXEC mode.

Verifying BFD Interoperability in a Cisco Nexus 9000 Series Device

The following example shows how to verify BFD interoperability in a Cisco Nexus 9000 Series device.

```
switch# show bfd neighbors details
OurAddr NeighAddr LD/RD RH/RS Holdown(mult) State Int
Vrf
10.1.1.1 10.1.1.2 1140850707/2147418093 Up 6393(4) Up Vlan2121
default
Session state is Up and using echo function with 50 ms interval
Local Diag: 0, Demand mode: 0, Poll bit: 0, Authentication: None
MinTxInt: 50000 us, MinRxInt: 2000000 us, Multiplier: 3
Received MinRxInt: 2000000 us, Received Multiplier: 4
Holdown (hits): 8000 ms (0), Hello (hits): 2000 ms (108)
Rx Count: 92, Rx Interval (ms) min/max/avg: 347/1996/1776 last: 1606 ms ago
Tx Count: 108, Tx Interval (ms) min/max/avg: 1515/1515/1515 last: 1233 ms ago
Registered protocols: ospf
Uptime: 0 days 0 hrs 2 mins 44 secs
Last packet: Version: 1 - Diagnostic: 0
State bit: Up - Demand bit: 0
Poll bit: 0 - Final bit: 0
Multiplier: 4 - Length: 24
My Discr.: 2147418093 - Your Discr.: 1140850707
Min tx interval: 2000000 - Min rx interval: 2000000
Min Echo interval: 1000 - Authentication bit: 0
Hosting LC: 10, Down reason: None, Reason not-hosted: None
```

```
switch# show bfd neighbors details
OurAddr NeighAddr LD/RD RH/RS Holdown(mult) State Int
Vrf
10.0.2.1 10.0.2.2 1140850695/131083 Up 270(3) Up Po14.121
default
Session state is Up and not using echo function
Local Diag: 0, Demand mode: 0, Poll bit: 0, Authentication: None
MinTxInt: 50000 us, MinRxInt: 50000 us, Multiplier: 3
Received MinRxInt: 100000 us, Received Multiplier: 3
Holdown (hits): 300 ms (0), Hello (hits): 100 ms (3136283)
Rx Count: 2669290, Rx Interval (ms) min/max/avg: 12/1999/93 last: 29 ms ago
Tx Count: 3136283, Tx Interval (ms) min/max/avg: 77/77/77 last: 76 ms ago
Registered protocols: ospf
Uptime: 2 days 21 hrs 41 mins 45 secs
Last packet: Version: 1 - Diagnostic: 0
State bit: Up - Demand bit: 0
Poll bit: 0 - Final bit: 0
Multiplier: 3 - Length: 24
My Discr.: 131083 - Your Discr.: 1140850695
Min tx interval: 100000 - Min rx interval: 100000
Min Echo interval: 0 - Authentication bit: 0
Hosting LC: 8, Down reason: None, Reason not-hosted: None
```

Verifying the BFD Configuration

To display BFD configuration information, perform one of the following:

Command	Purpose
show running-config bfd	Displays the running BFD configuration.

Command	Purpose
show startup-config bfd	Displays the BFD configuration that will be applied on the next system startup.

Monitoring BFD

Use the following commands to display BFD:

Command	Purpose
show bfd neighbors [**application** *name*] [**details**]	Displays information about BFD for a supported application, such as BGP or OSPFv2.
show bfd neighbors [**interface** *int-if*] [**details**]	Displays information about BFD neighbors on an interface.
show bfd neighbors [**dest-ip** *ip-address*] [**src-ip** *ip-address*][**details**]	Displays information about the specified BFD neighbors on an interface.
show bfd neighbors [**vrf** *vrf-name*] [**details**]	Displays information about BFD for a VRF.
show bfd [**ipv4** | **ipv6**] [**neighbors**]	Displays information about IPv4 neighbors or IPv6 neighbors.

BFD Multihop

BFD multihop for IPv4 and BFD multihop for IPv6 are supported in compliance with RFC5883. BFD multihop sessions are set up between a unique source and destination address pair. A multihop BFD session is associated with the link between a source and destination rather than with an interface, as with single-hop BFD sessions.

BFD Multihop Number of Hops

BFD multihop sets the TTL field to the maximum limit, and it does not check the value on reception. The BFD code has no impact on the number of hops a BFD multihop packet can traverse. However, in most of the systems, it limits the number of hops to 255.

Guidelines and Limitations for BFD Multihop

BFD multihop has the following configuration guidelines and limitations:

- Beginning with Cisco NX-OS Release 9.3(6), BFD multihop is only supported in BGP IPv4 on Cisco Nexus 9200, 9300-EX/FX/GX platform switches and Cisco Nexus 9500 platform switches with N9K-X9700-EX line cards.
- In a dynamic BGP configuration, both the single and multihop BGP peers accepts BFD multihop configuration.
- BFD multihop is only supported with BGP.

- BFD multihop is supported for BGP IPv6 multihop neighbors on the following devices:
  - Cisco Nexus 9200YC-X, 9300-EX, 9300-FX and 9300-GX switches
  - Cisco Nexus 9500 platform switches with N9K-X9736C-EX, N9K-X97160YC-EX, N9K-X9732C-EX, N9K-X9732C-EXM, or N9K-X9736C-FX line cards

Note You must enable the **system routing template-mpls-heavy** command in order to use BFD multihop for BGP IPv6 with Cisco Nexus 9500 platform switches with -EX and -FX line cards.

- Multihop BFD is identified with UDP Destination port 4784.
- The default interval timer for multihop BFD is 250 ms with multiplier 3.
- The maximum number of supported multihop BFD sessions is 100.
- Existing BFD authentication support is extended for multihop sessions.
- Echo mode is not supported for multihop BFD.
- Multihop with segment routing underlay is not supported.
- On unsupported platforms, BFD commands are accepted when configuring BGPv6 multihop neighbors. However, the sessions will not be created or installed.
- When Multihop BFD session is installed in port-channel, the following points must be taken care:
  - If all the sessions are hosted on a single line card of Cisco Nexus 9500 family switches, during reloading of hosted line cards all the sessions will be hosted on another line card. BFD and BGP sessions may flap in this case.
  - Multihop BFD session for BGP over cross modules port-channel doesn't provide full redundancy.

Configuring BFD Multihop Session Global Interval Parameters

You can configure the BFD session global parameters for all BFD sessions on the device. Different BFD session parameters for each session can be achieved using the per session configuration commands .

Before you begin

Enable the BFD feature.

SUMMARY STEPS

1. **configure terminal**
2. **[no] bfd multihop interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier*
3. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**[no] bfd multihop interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier* **Example:** `switch(config)# bfd multihop interval 250 min_rx 250 multiplier 3`	Configures the BFD multihop session global parameters for all BFD sessions on the device. This command overrides the default values. The *Required Minimum Receive Interval* and *Desired Minimum Transmit Interval* are 250. The multiplier default is 3.
Step 3	**end** **Example:** `switch(config)# end`	Saves the configuration change and ends the configuration session.

Configuring Per Multihop Session BFD Parameters

You can configure per multihop session BFD parameters.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *as-number*
3. **neighbor** (*ip-address* | *ipv6-address*) **remote-as** *as-number*
4. **update-source** *interface*
5. **bfd**
6. **bfd multihop interval** *mintx* **min_rx** *msec* **multiplier** *value*
7. **bfd multihop authentication keyed-sha1 keyid** *id* **key** *ascii_key*
8. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters configuration mode.

Step 2	**router bgp** *as-number*
	Example:
	`switch(config)# router bgp 64496`
	`switch(config-router)#`
		Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 3	**neighbor** (*ip-address* | *ipv6-address*) **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 209.165.201.1 remote-as 64497`
	`switch(config-router-neighbor)#`
		Configures the IPv4 or IPv6 address and AS number for a remote BGP peer. The *ip-address* format is x.x.x.x. The *ipv6-address* format is A:B::C:D.

Step 4	**update-source** *interface*
	Example:
	`switch(config-router-neighbor)# update-source Ethernet1/4`
	`switch(config-router-neighbor)#`
		Retrieves the source IP address of the BFD session from the interface.

Step 5	**bfd** **Example:** `switch(config-router-neighbor)# bfd multihop`	Enables BFD for this BGP peer.

Step 6	**bfd multihop interval** *mintx* **min_rx** *msec* **multiplier** *value*
	Example:
	`switch(config-router-neighbor)# bfd multihop interval 250`
	`min_rx 250 multiplier 3`
		Configures Multihop BFD interval values for this neighbor. The *mintx* and *msec* range is from 250 to 999 milliseconds and the default is 250. The multiplier range is from 1 to 50. The multiplier default is 3.

Step 7	**bfd multihop authentication keyed-sha1 keyid** *id* **key** *ascii_key*
	Example:
	`switch(config-router-neighbor)# bfd multihop authentication`
	`keyed-sha1 keyid 1 ascii_key cisco123`
		(Optional) Configures SHA-1 authentication for BFDs on Multihop BFD session over this neighbor. The *ascii_key* string is a secret key shared among BFD peers. The *id* value, a number between 0 and 255, is assigned to this particular *ascii_key*. BFD packets specify the key by *id*, allowing the use of multiple active keys.
		To disable SHA-1 authentication on the interface, use the **no** form of the command.

Step 8	**copy running-config startup-config** **Example:**	(Optional) Saves the configuration change.

	Command or Action	Purpose
	`switch(config-router-neighbor)# copy running-config startup-config`	

Configuration Examples for BFD

This example shows how to configure BFD for OSPFv2 on Ethernet 2/1, using the default BFD session parameters:

```
feature bfd
feature ospf
router ospf Test1
interface ethernet 2/1
ip ospf bfd
no shutdown
```

This example shows how to configure BFD for all EIGRP interfaces, using the default BFD session parameters:

```
feature bfd
feature eigrp
bfd interval 100 min_rx 100 multiplier 4
router eigrp Test2
bfd
```

This example shows how to configure BFDv6:

```
feature bfd
feature ospfv3
router ospfv3 Test1
interface Ethernet2/7
  ipv6 router ospfv3 Test1 area 0.0.0.0
  ospfv3 bfd
  no shutdown
```

Show Example for BFD

This example shows results of the **show bfd ipv6 neighbors details** command.

```
#show bfd ipv6 neighbors details

OurAddr                          NeighAddr
 LD/RD                 RH/RS           Holdown(mult)     State       Int
 Vrf
cc:10::2                         cc:10::1
 1090519335/1090519260 Up              5692(3)           Up          Po1
 default


Session state is Up and using echo function with 250 ms interval
Local Diag: 0, Demand mode: 0, Poll bit: 0, Authentication: None
MinTxInt: 250000 us, MinRxInt: 2000000 us, Multiplier: 3
Received MinRxInt: 2000000 us, Received Multiplier: 3
```

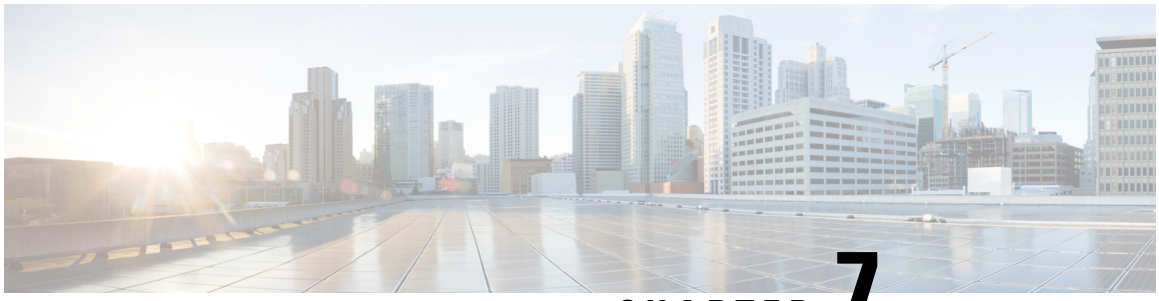
```
Holdown (hits): 6000 ms (4), Hello (hits): 2000 ms (205229)
Rx Count: 227965, Rx Interval (ms) min/max/avg: 124/1520/1510 last: 307 ms ago
Tx Count: 205229, Tx Interval (ms) min/max/avg: 1677/1677/1677 last: 587 ms ago
Registered protocols:  bgp
Uptime: 3 days 23 hrs 31 mins 13 secs
Last packet: Version: 1               - Diagnostic: 0
             State bit: Up            - Demand bit: 0
             Poll bit: 0              - Final bit: 0
             Multiplier: 3            - Length: 24
             My Discr.: 1090519260    - Your Discr.: 1090519335
             Min tx interval: 250000  - Min rx interval: 2000000
             Min Echo interval: 250000 - Authentication bit: 0
Hosting LC: 1, Down reason: None, Reason not-hosted: None
```

Related Documents

Related Topic	Document Title
BFD commands	*Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide*

RFCs

RFC	Title
RFC 5880	*Bidirectional Forwarding Detection (BFD)*
RFC 5881	*BFD for IPv4 and IPv6 (Single Hop)*
RFC 7130	*Bidirectional Forwarding Detection (BFD) on Link Aggregation Group (LAG) Interfaces*



CHAPTER 7

Configuring Port Channels

About Port Channels

A port channel is an aggregation of multiple physical interfaces that creates a logical interface. You can bundle up to 32 individual active links into a port channel 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 can create a Layer 2 port channel by bundling compatible Layer 2 interfaces, or you can create Layer 3 port channels by bundling compatible Layer 3 interfaces. You cannot combine Layer 2 and Layer 3 interfaces in the same port channel.

You can also change the port channel from Layer 3 to Layer 2. See the Configuring Layer 2 Interfaces chapter for information about creating Layer 2 interfaces.

A Layer 2 port channel interface and it's member ports can have different STP parameters. Changing the STP parameters of the port channel does not impact the STP parameters of the member ports because a port channel interface takes precedence if the member ports are bundled.

Note After a Layer 2 port becomes part of a port channel, all switchport configurations must be done on the port channel; you can no longer apply switchport configurations to individual port-channel members. You cannot apply Layer 3 configurations to an individual port-channel member either; you must apply the configuration to the entire port channel.

In releases prior to Cisco NX-OS Release 9.3(7), in a port-channel configuration with a member port operating as an individual (I), you can define the STP port-type under the member port rather than the port-channel.

Beginning with Cisco NX-OS Release 9.3(7), in a port-channel configuration with a member port operating as an individual (I), you can no longer define the STP port-type under the member port. It remains blocked by the STP. You must configure the STP port-type under the port-channel.

You can use static port channels, with no associated aggregation protocol, for a simplified configuration.

For more flexibility, 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. You cannot configure LACP on shared interfaces.

See the LACP Overview section for information about LACP.

Port Channels

A port channel bundles physical links into a channel group to create a single logical link that provides the aggregate bandwidth of up to 32 physical links. If a member port within a port channel fails, the traffic previously carried over the failed link switches to the remaining member ports within the port channel.

However, you can enable the LACP to use port channels more flexibly. Configuring port channels with LACP and static port channels require a slightly different procedure (see the “Configuring Port Channels” section).

Note The device does not support Port Aggregation Protocol (PAgP) for port channels.

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 duplex mode (see the “Compatibility Requirements” section). When you run static port channels with no aggregation protocol, the physical links are all in the on channel mode; you cannot change this mode without enabling LACP (see the “Port-Channel Modes” section).

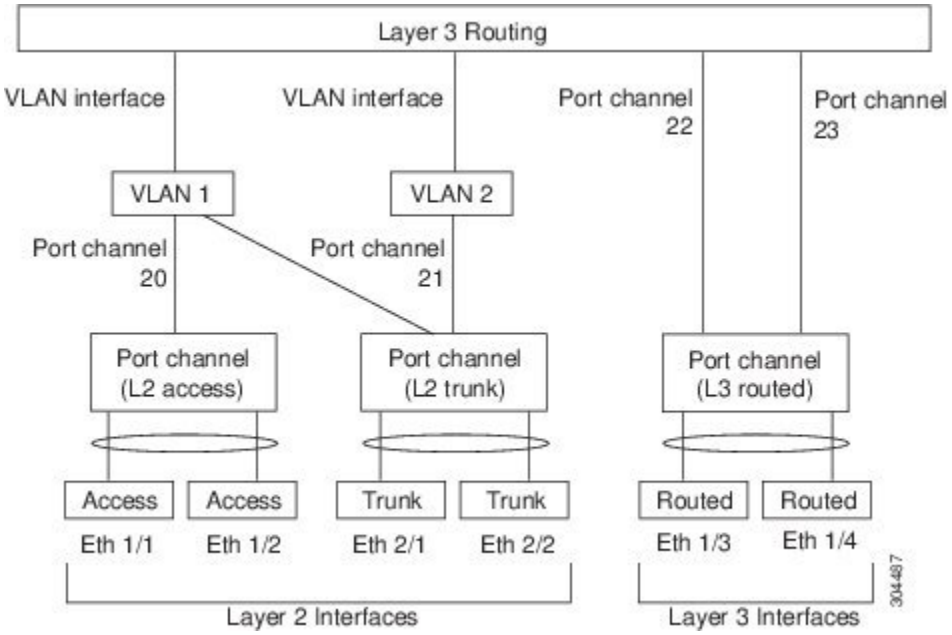
You can create port channels directly by creating the port-channel interface, or you can create a channel group that acts to aggregate individual ports into a bundle. When you associate an interface with a channel group, the software creates a matching port channel automatically if the port channel does not already exist. In this instance, the port channel assumes the Layer 2 or Layer 3 configuration of the first interface. You can also create the port channel first. In this instance, the Cisco NX-OS software creates an empty channel group with the same channel number as the port channel and takes the default Layer 2 or Layer 3 configuration, as well as the compatibility configuration (see the “Compatibility Requirements” section).

Note The 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.

Port-Channel Interfaces

The following shows port-channel interfaces.

Figure 9: Port-Channel Interfaces



You can classify port-channel interfaces as Layer 2 or Layer 3 interfaces. In addition, you can configure Layer 2 port channels in either access or trunk mode. Layer 3 port-channel interfaces have routed ports as channel members.

You can configure a Layer 3 port channel with a static MAC address. If you do not configure this value, the Layer 3 port channel uses the router MAC of the first channel member to come up. See the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide for information about configuring static MAC addresses on Layer 3 port channels.

See the "Configuring Layer 2 Interfaces" chapter for information about configuring Layer 2 ports in access or trunk mode and the "Configuring Layer 3 Interfaces" chapter for information about configuring Layer 3 interfaces and subinterfaces.

Basic Settings

You can configure the following basic settings for the port-channel interface:

- Bandwidth—Use this setting for informational purposes only; this setting is to be used by higher-level protocols.
- Delay—Use this setting for informational purposes only; this setting is to be used by higher-level protocols.
- Description
- Duplex

- IP addresses
- Maximum Transmission Unit (MTU)
- Shutdown
- Speed

Compatibility Requirements

When you add an interface to a channel group, the software checks certain interface attributes to ensure that the interface is compatible with the channel group. For example, you cannot add a Layer 3 interface to a Layer 2 channel group. The Cisco NX-OS software 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:

- Network layer
- (Link) speed capability
- Speed configuration
- Duplex capability
- Duplex configuration
- Port mode
- Access VLAN
- Trunk native VLAN
- Tagged or untagged
- Allowed VLAN list
- MTU size
- SPAN—Cannot be a SPAN source or a destination port
- Storm control
- Flow-control capability
- Flow-control configuration
- Media type, either copper or fiber

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

You can only add interfaces configured with the channel mode set to on to static port channels, and you can 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. If you configure a member port with an incompatible attribute, the software suspends that port in the port channel.

Alternatively, you can force ports with incompatible parameters to join the port channel if the following parameters are the same:

- (Link) speed capability
- Speed configuration
- Duplex capability
- Duplex configuration
- Flow-control capability
- Flow-control configuration

When the interface joins a port channel, some of its individual parameters are removed and replaced with the values on the port channel as follows:

- Bandwidth
- Delay
- Extended Authentication Protocol over UDP
- VRF
- IP address
- MAC address
- Spanning Tree Protocol
- NAC
- Service policy
- Access control lists (ACLs)

Many interface parameters remain unaffected when the interface joins or leaves a port channel as follows:

- Beacon
- Description
- CDP
- LACP port priority
- Debounce
- UDLD
- MDIX
- Rate mode
- Shutdown
- SNMP trap

Note When you delete the port channel, the software sets all member interfaces as if they were removed from the port channel.

See the “LACP Marker Responders” section for information about port-channel modes.

Load Balancing Using Port Channels

The Cisco NX-OS software load balances traffic across all operational interfaces in a port channel by hashing 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. Port-channel load balancing uses MAC addresses, IP addresses, or Layer 4 port numbers to select the link. Port-channel load balancing uses either source or destination addresses or ports, or both source and destination addresses or ports.

You can configure the load- balancing mode to apply to all port channels that are configured on the entire device. You can configure one load-balancing mode for the entire device. You cannot configure the load-balancing method per port channel.

You can configure the type of load-balancing algorithm used. You can choose the load-balancing algorithm that determines which member port to select for egress traffic by looking at the fields in the frame.

The default load-balancing mode for Layer 3 interfaces is the source and destination IP L4 ports, and the default load-balancing mode for non-IP traffic is the source and destination MAC address. Use the **port-channel load-balance** command to set the load-balancing method among the interfaces in the channel-group bundle. The default method for Layer 2 packets is src-dst-mac. The default method for Layer 3 packets is src-dst ip-l4port.

You can configure the device to use one of the following methods 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
- Source TCP/UDP port number
- Destination TCP/UDP port number
- Source and destination TCP/UDP port number
- GRE inner IP headers with source, destination and source-destination

Non-IP and Layer 3 port channels both follow the configured load-balancing method, using the source, destination, or source and destination parameters. For example, when you configure load balancing to use the source IP address, all non-IP traffic uses the source MAC address to load balance the traffic while the Layer 3 traffic load balances the traffic using the source IP address. Similarly, when you configure the destination

MAC address as the load-balancing method, all Layer 3 traffic uses the destination IP address while the non-IP traffic load balances using the destination MAC address.

The unicast and multicast traffic is load-balanced across port-channel links based on configured load-balancing algorithm displayed 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 Devices that run Cisco IOS can optimize the behavior of the member ports ASICs if a failure of a single member occurred by running the port-channel hash-distribution command. The Cisco Nexus 9000 Series device performs this optimization by default and does not require or support this command. Cisco NX-OS does support the customization of the load-balancing criteria on port channels through the port-channel load-balance command for the entire device.

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.

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.

Only the following load-balancing algorithms support symmetric hashing:

- src-dst ip
- src-dst ip-l4port

Guidelines and Limitations for ECMP

You might observe that load balancing with Layer 2/Layer 3 GW flows are not load balanced equally among all links when the switch comes up initially after reload. There are two CLIs to change the ECMP hash configuration in the hardware. The two CLI commands are mutually exclusive.

- Enter the **port-channel load-balance [src | src-dst | dst] mac** command for MAC-based only hash.
- For hash based on IP/Layer 4 ports, enter either the **ip load-share** or **port-channel load-balance** command.

- The **port-channel load-balance** command can overwrite the **ip load-share** command. It is better to enter the **port-channel load-balance** command which helps to set both the IP and MAC parameters.
- There are no options to force the hashing algorithm based on the IP/Layer 4 port. The default MAC configuration is always programmed as a part of the port channel configuration.
- ECMP resilient hashing is not supported for traffic flows over tunnel.

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 maps flows to physical ports and it is supported for both ECMP groups and port channel interfaces.

If a physical link fails, the flows originally assigned to the failed link are redistributed uniformly among the remaining working links. The existing flows through the working links are not rehashed and hence are not impacted.

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

Resilient hashing is supported on all the Cisco Nexus 9000 Series platforms . Beginning Cisco NX-OS Release 9.3(3), resilient hashing is supported on Cisco Nexus 92160YC-X, 92304QC, 9272Q, 9232C, 9236C, 92300YC switches.

GTP Tunnel Load Balancing

Introduction

GPRS Tunneling Protocol (GTP) is used mainly to deliver mobile data on wireless networks via Cisco Nexus 9000 Series switches as the core router. When two routers carrying GTP traffic are connected with link bundling, the traffic is required to be distributed evenly between all bundle members.

Different Mechanisms for GTP Load Balancing

Two different kinds of mechanisms are used to achieve GTP load balancing.

- From Cisco Nexus Release 10.5(2), the inner IP header fields source, destination IP address and IP protocol is used to maintain load balancing.
- Prior to Cisco Nexus Release 10.5(2), the 5-tuple load balancing mechanism is used. The load balancing mechanism takes into account the source IP, destination IP, protocol, Layer 4 resource and destination

port (if traffic is TCP or UDP) fields from the packet. In the case of GTP traffic, a limited number of unique values for these fields restrict the equal distribution of traffic load on the tunnel.

Inner IP Header GTP Load Balancing Mechanism

Using inner IP header fields source-ip, dest-ip and ip-protocol the load-balancing is done. Symmetric load-balancing is supported to maintain stickiness for forward and reverse traffic of same flow.

GTP inner header based hashing works for both IPv4 and IPv6 on all interfaces. The inner IP header for both IPv4 and IPv6 uses all the 16 UDF for all cloudscale switches. Inner IP headers are used for two switch or three switches bundling.

5-Tuple GTP Load Balancing Mechanism

In order to avoid polarization for GTP traffic in load balancing, a tunnel endpoint identifier (TEID) in the GTP header is used instead of a UDP port number. Since the TEID is unique per tunnel, traffic can be evenly load balanced across multiple links in the bundle.

This feature overrides the source and destination port information with the 32-bit TEID value that is present in GTPU packets.

GTP tunnel load balancing feature adds support for:

- GTP with IPv4/IPv6 transport header on physical interface
- GTP traffic over TE tunnel
- GTPU with UDP port 2152

The **ip load-sharing address source-destination gtpu** command enables the GTP tunnel load balancing.

To know the egress interface for GTP traffic after load balancing, use **show cef {ipv4 | ipv6} exact-route** command with TEID in place of L4 protocol source and destination port number. Use 16MSBist of TEID in source port and 16LSBits of TEID in destination port.

The **port-channel load-balance src-dst gtpu** command enables GTP packets with UDP destination port number 2152 to load balance based on the GTP TEID value. This command enables the switch to load balance for GTP packets even if the outer five tuples (*src-ip*, *dst-ip*, *ip proto*, *L4 sport*, *L4 dport*) are same. Because the hardware controls for port channel and ECMP are same, enabling either port-channel load-balance or ip load-sharing with GTP option enables GTP TEID based load balancing.

- The **port-channel load-balance src-dst gtpu** command is applicable for both GTP packets, with or without VXLAN encapsulation
- When GTP header is a part of the outer layer, the **port-channel load-balance src-dst gtpu** command picks up GTP TEID from outer layer for hashing.
- When GTP header is part of inner layer, the **port-channel load-balance src-dst gtpu** command picks up GTP TEID from inner layer for hashing.

You need to set the protocol field to 17 and set the value for other parameters when you use the **show port-channel load-balance forwarding-path** command. An example is listed below.

```
switch(config)# show port-channel load-balance forwarding-path interface port-channel 2
src-ip 1.1.1.1 dst-ip 2.2.2.2 gtpteid
0x3 protocol 17
```

Supported Platforms

Beginning Cisco Nexus Release 9.3(3) GTP Tunnel Load Balancing is supported on Cisco Nexus 9500 platform switches with 9700-EX and 9700-FX line cards. However, GTP Tunnel Load Balancing for IPv6 flow is supported only on Cisco Nexus 9500 platform switches with FM-E2 fabric modules. It is not supported on Cisco Nexus 9500 platform switches with FM-E fabric modules. Because the hardware control is same for both Port-channel and ECMP, enabling either port-channel load-balance or ip load-sharing with GTP option enables GTP TEID based load balancing for both the cases. In multi encapsulated packets, if the GTP header is a part of outer header, it picks up GTP TEIF from outer layer for hashing. If the GTP header is a part of inner header, it picks up GTP TEIF from inner layer for hashing.

GTP Tunnel Load Balancing is supported on Cisco Nexus 9300-EX, 9300-FX, 9300-FX2, 9364C, and 9300-GX platform switches.

Inner IP header GTP load balancing mechanism is supported on:

- Cisco Nexus 9300-EX platform switches
- Cisco Nexus 9300-FX and 9364C platform switches
- Cisco Nexus 9500 platform switches with 9700-EX and 9700-FX line cards
- Cisco Nexus 9300-EX, 9300-FX, 9300-FX2, 9364C, and 9300-GX platform switches
- Cisco Nexus 9364C-H1 Switch

Note Cisco Nexus 9364C-H1 switch can natively support inner-header based hashing for packets with GTP header of size 8 or 12 bytes

LACP

LACP allows you to configure up to 16 interfaces into a port channel.

LACP Overview

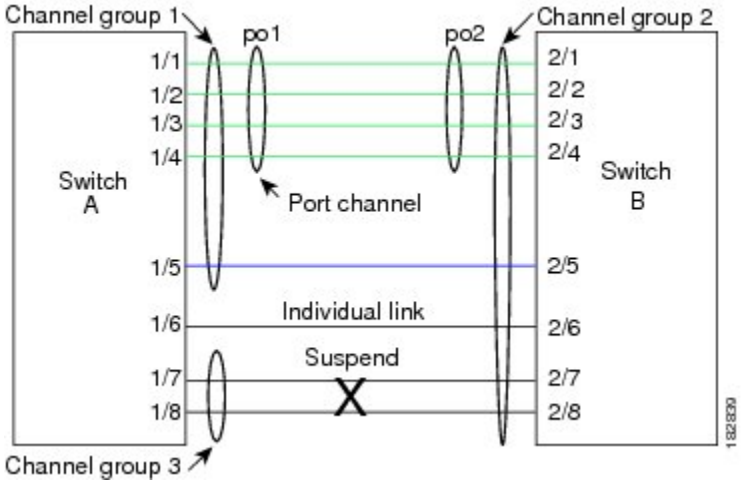
The Link Aggregation Control Protocol (LACP) for Ethernet is defined in IEEE 802.1AX and IEEE 802.3ad. This protocol controls how physical ports are bundled together to form one logical channel.

Note You must enable LACP before you can use LACP. By default, LACP is disabled. See the “Enabling LACP” section for information about enabling LACP.

The system automatically takes a checkpoint before disabling the feature, and you can roll back to this checkpoint. See the Cisco Nexus 9000 Series NX-OS System Management Configuration Guide for information about rollbacks and checkpoints.

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 10: Individual Links Combined into a Port Channel



With LACP, you can bundle up to 32 interfaces in a channel group.

Note When you delete the port channel, the software automatically deletes the associated channel group. All member interfaces revert to their original configuration.

Note If you downgrade a Cisco Nexus 9500 series switch that is configured to use LACP vPC convergence feature, that runs Cisco NX-OS Release 7.0(3)I7(5) to a lower release, the configuration is removed. You must configure the LACP vPC convergence feature again when you upgrade the switch.

You cannot disable LACP while any LACP configurations are present.

Port-Channel Modes

Individual interfaces in port channels are configured with channel modes. When you run static port channels with no aggregation 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 either **active** or **passive**. You can configure channel mode for individual links in the LACP channel group when you are adding the links to the 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 14: Channel Modes for Individual Links in a Port Channel

Channel Mode	Description
passive	The LACP is enabled on this port channel and the ports are in a passive negotiating state. Ports responds to LACP packets that it receives but does not initiate LACP negotiation.
active	The LACP is enabled on this port channel and the ports are in an active negotiating state. Ports initiate negotiations with other ports by sending LACP packets.
on	The LACP is disabled on this port channel and the ports are in a non-negotiating state. The **on** state of the port channel represents the static mode. The port will not verify or negotiate port channel memberships. If you attempt to change the channel mode to active or passive before enabling LACP, the device displays an error message. 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. The **on** state is the default port-channel mode

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.

Two devices can form an LACP port channel when their ports are in different LACP modes if the modes are compatible as in the following example:

Table 15: Channel Modes Compatibility

Device 1 > Port-1	Device 2 > Port-2	Result
Active	Active	Can form a port channel.
Active	Passive	Can form a port channel.
Passive	Passive	Cannot form a port channel because no ports can initiate negotiation.
On	Active	Cannot form a port channel because LACP is enabled only on one side.
On	Passive	Cannot form a port channel because LACP is not enabled.

LACP ID Parameters

This section describes the LACP 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 that is 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 and the duplex capability
- Configuration restrictions that you establish

LACP Marker Responders

You can dynamically redistribute the data traffic by using port channels. This redistribution might result from a removed or added link or a change in the load-balancing scheme. Traffic redistribution that occurs in the middle of a traffic flow can cause misordered frames.

LACP uses the Marker Protocol to ensure that frames are not duplicated or reordered due to this redistribution. The Marker Protocol detects when all the frames of a given traffic flow are successfully received at the remote end. LACP sends Marker PDUs on each of the port-channel links. The remote system responds to the Marker PDU once it receives all the frames received on this link prior to the Marker PDU. The remote system then sends a Marker Responder. Once the Marker Responders are received by the local system on all member links of the port channel, the local system can redistribute the frames in the traffic flow with no chance of misordering. The software supports only Marker Responders.

LACP-Enabled and Static Port Channels Differences

The following table summarizes the major differences between port channels with LACP enabled and static port channels.

Table 16: 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: • Active • Passive	Can only be On
Maximum number of links in channel	32	32

LACP Compatibility Enhancements

When a Cisco Nexus 9000 Series device is connected to a non-Nexus peer, its graceful failover defaults may delay the time that is taken to bring down a disabled port or cause traffic from the peer to be lost. To address these conditions, the **lacp graceful-convergence** command was added.

By default, LACP sets a port to suspended state if it does not receive an LACP PDU from the peer. **lacp suspend-individual** is a default configuration on Cisco Nexus 9000 series switches. This command puts the port in suspended state if it does not receive any LACP PDUs. In some cases, although this feature helps in preventing loops created due to misconfigurations, it can cause servers fail to boot up because they require LACP to logically bring up the port. You can put a port into an individual state by using the **no lacp suspend-individual**. Port in individual sate takes attributes of the individual port based on the port configuration.

LACP port-channels exchange LACP PDUs for quick bundling of links when connecting a server and a switch. However, the links go into suspended state when the PDUs are not received.

The **delayed LACP** feature enables one port-channel member, the delayed-LACP port, to come up first as a member of a regular port-channel before LACP PDUs are received. After it is connected in LACP mode, other members, the auxiliary LACP ports, are brought up. This avoids having the links becoming suspended when PDUs are not received.

Which port in the port-channel comes up first depends on the port-priority value of the ports. A member link in a port channel with lowest priority value, will come come up first as a LACP delayed port. Regardless of the operational status of the links, the configured priority of a LACP port is used to select the delayed-lacp port

Guidelines and Limitations

This feature supports Layer 2 port-channels with or without VPC running in spanning-tree port type trunk mode. These guidelines and limitations apply to LACP:

- Using **no lacp suspend-individual** and **lacp mode delay** on a same port channel is not recommended because it can put non-lacp delayed ports in individual state. As a best practice, you must avoid combining these two configurations.
- Not supported on Layer 3 port-channels.
- Not supported on Nexus 9000 switches on the FEX NIF fabric port-channel or FEX HIF host port-channels

LACP Port-Channel Minimum Links and LACP MaxBundle

A port channel aggregates similar ports to provide increased bandwidth in a single manageable interface.

The introduction of the minimum links and LACP MaxBundle feature further refines LACP port-channel operation and provides increased bandwidth in one manageable interface.

The LACP port-channel minimum links feature does the following:

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

The LACP MaxBundle defines the maximum number of bundled ports allowed in a LACP port channel.

The LACP MaxBundle feature does the following:

- Defines an upper limit on the number of bundled ports in an LACP port channel.
- Allows hot-standby ports with fewer bundled ports. (For example, in an LACP port channel with five ports, you can designate two of those ports as hot-standby ports.)

Note The minimum links and LACP MaxBundle features only work with LACP port-channels. The switch allows you to configure these features on non-LACP port-channels, but the features are not operational.

LACP Fast Timers

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. To configure the LACP fast time rate, see the “Configuring the LACP Fast Timer Rate” section.

ISSU and ungraceful switchovers are not supported with LACP fast timers.

Virtualization Support

You must configure the member ports and other port channel-related configuration from the virtual device context (VDC) that contains the port channel and member ports. You can use the numbers from 1 to 4096 in each VDC to number the port channels.

All ports in one port channel must be in the same VDC. When you are using LACP, all possible 8 active ports and all possible 8 standby ports must be in the same VDC.

Note You must configure load balancing using port channels in the default VDC. See the "Load Balancing Using Port Channels" section for more information about load balancing.

High Availability

Port channels provide high availability by load balancing traffic across multiple ports. If a physical port fails, the port channel is still operational if there is an active member in the port channel. You can bundle ports from different modules and create a port channel that remains operational even if a module fails because the settings are common across the module.

Port channels support stateful and stateless restarts. A stateful restart occurs on a supervisor switchover. After the switchover, the Cisco NX-OS software applies the runtime configuration after the switchover.

The port channel goes down if the operational ports fall below the configured minimum links number.

Note See the *Cisco Nexus 9000 Series NX-OS High Availability and Redundancy Guide* for complete information about high-availability features.

Prerequisites for Port Channeling

Port channeling has the following prerequisites:

- You must be logged onto the device.
- All ports for a single port channel must be either Layer 2 or Layer 3 ports.
- All ports for a single port channel must meet the compatibility requirements. See the "Compatibility Requirements" section for more information about the compatibility requirements.
- You must configure load balancing from the default VDC.

Guidelines and Limitations

Port channeling has the following configuration guidelines and limitations:

- For scaled port-channel deployments on Cisco Nexus 9516 switch with Gen 1 line cards, you need to use the **port-channel scale-fanout** command followed by **copy run start** and **reload** commands.
- **show** commands with the **internal** keyword are not supported.
- The LACP port-channel minimum links and maxbundle feature is not supported for host interface port channels.
- Enable LACP before you can use that feature.

- You can configure multiple port channels on a device.
- Do not put shared and dedicated ports into the same port channel. (See the "Configuring Basic Interface Parameters" chapter for information about shared and dedicated ports.)
- For Layer 2 port channels, ports with different STP port path costs can form a port channel if they are compatibly configured with each other. See the "Compatibility Requirements" section for more information about the compatibility requirements.
- When L2 ePBR is configured between the L3 port channel interface, port channel will not come up as the LACP packet drops at the ePBR device.
- In STP, the port-channel cost is based on the aggregated bandwidth of the port members.
- After you configure a port channel, the configuration that you apply to the port channel interface affects the port channel member ports. The configuration that you apply to the member ports affects only the member port where you apply the configuration.
- LACP does not support half-duplex mode. Half-duplex ports in LACP port channels are put in the suspended state.
- Do not configure ports that belong to a port channel group as private VLAN ports. While a port is part of the private VLAN configuration, the port channel configuration becomes inactive.
- Channel member ports cannot be a source or destination SPAN port.
- Port-channels are not supported on generation 1 100G line cards (N9K-X9408PC-CFP2) or generic expansion modules (N9K-M4PC-CFP2).
- Port-channels are supported on devices with generation 2 (and later) 100G interfaces.
- The port channel might be affected by the limitations of the Application Leaf Engine (ALE) uplink ports on Cisco Nexus 9300 and 9500 Series devices:Limitations for ALE Uplink Ports.
- Resilient hashing (port-channel load-balancing resiliency) and VXLAN configurations are not compatible with VTEPs using ALE uplink ports.

Note Resilient hashing is disabled by default.

- The maximum number of subinterfaces for a satellite/FEX port is 63.
- On a Cisco Nexus 92300YC switch, the first 24 ports that are part of the same quadrant. All the ports in the same quadrant must have same speed. Having different speed on ports in a quadrant is not supported. Following are the first 24 ports on the Cisco Nexus 92300YC switch that share same quadrant:
  - 1,4,7,10
  - 2,5,8,11
  - 3,6,9,12
  - 13,16,19,22
  - 14,17,20,23
  - 15,18,21,24

- On a Cisco Nexus 9500 switch with a X96136YC-R line card, the ports 17–48 are part of the same quadrant. Ports in the same quadrant must have same speed (1/10G or 25G) on all ports. Having different speed on ports in a quadrant is not supported. If you set different speed in any of the ports in a quadrant, the ports go into error disable state. Interfaces in same quadrant are:
  - 17–20
  - 21–24
  - 25–28
  - 29–32
  - 33–36
  - 37–40
  - 41–44
  - 45–48
- Resilient hashing is supported on Cisco Nexus 9500 Series switches with N9K-X9636C-R, N9K-X9636Q-R, N9K-X9636C-RX, and N9K-X96136YC-R line cards.
- Port-channel symmetric hashing is supported on Cisco Nexus 9200, 9300-EX, 9300-FX/FX2, and 9300-GX platform switches and Cisco Nexus 9500 platform switches with N9K-X9732C-EX, N9K-X9736C-EX, N9K-X9736C-FX, and N9K-X9732C-FX line cards.
- ECMP symmetric hashing is supported on Cisco Nexus 9200, 9300-EX, and 9300-FX/FX2 platform switches and Cisco Nexus 9500 platform switches with N9K-X9732C-EX, N9K-X9736C-EX, N9K-X9736C-FX, and N9K-X9732C-FX line cards.
- GRE inner headers are supported on the following switches:
  - Cisco Nexus 9364C platform switches
  - Cisco Nexus 9336C-FX2, 9348GC-FXP, 93108TC-FX, 93180YC-FX, and 93240YC-FX2 platform switches
  - Cisco Nexus 9300-GX platform switches.
  - Cisco Nexus 9500 platform switches with N9K-X9736C-FX line cards
- Beginning with Cisco NX-OS Release 9.3(6), Cisco Nexus 9300-FX2 platform switches support the coexistence of VXLAN and IP-in-IP tunneling. For more information, including limitations, see the **VXLAN and IP-in-IP Tunneling** section in the *Cisco Nexus 9000 Series NX-OS VXLAN Configuration Guide, Release 9.3(x)*.
- For FEX interfaces using LACP, all DME oper/runtime properties for the FEX interfaces does not get updated. All runtime updates for FEX ports happens from FEX LACP process context and are not communicated to the parent switch.This is a day-1 behaviour.
- Beginning with Cisco NX-OS Release 10.3(1)F, the hashing based on src/dst ip and src/dst L4 port number is supported on Cisco Nexus 9808 platform switches.

Default Settings

The following table lists the default settings for port-channel parameters.

Table 17: Default Port-Channel Parameters

Parameters	Default
Port channel	Admin up
Load balancing method for Layer 3 interfaces	Source and destination IP address
Load balancing method for Layer 2 interfaces	Source and destination MAC address
Load balancing per module	Disabled
LACP	Disabled
Channel mode	on
LACP system priority	32768
LACP port priority	32768
Minimum links for LACP	1
Maxbundle	32
Minimum links for FEX fabric port channel	1

Configuring Port Channels

Note See the "Configuring Basic Interface Parameters" chapter for information about configuring the maximum transmission unit (MTU) for the port-channel interface. See the "Configuring Layer 3 Interfaces" chapter for information about configuring IPv4 and IPv6 addresses on the port-channel interface.

Note If you are familiar with the Cisco IOS CLI, be aware that the Cisco NX-OS commands for this feature might differ from the Cisco IOS commands that you would use.

Creating a Port Channel

You can create a port channel before you create a channel group. The software automatically creates the associated channel group.

Note When the port channel is created before the channel group, the port channel should be configured with all of the interface attributes that the member interfaces are configured with. Use the **switchport mode trunk** {*allowed vlan vlan-id* | *native vlan-id*} command to configure the members.

This is required only when the channel group members are Layer 2 ports (switchport) and trunks (switchport mode trunk).

Note Use the **no interface port-channel** command to remove the port channel and delete the associated channel group.

Command	Purpose
no interface port-channel *channel-number* **Example:** `switch(config)# no interface port-channel 1`	Removes the port channel and deletes the associated channel group.

Before you begin

Enable LACP if you want LACP-based port channels.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-number*
3. **show port-channel summary**
4. **no shutdown**
5. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *channel-number*
	Example:
	`switch(config)# interface port-channel 1`
	`switch(config-if)`
		Specifies the port-channel interface to configure, and enters the interface configuration mode. The range is from 1 to 4096. The Cisco NX-OS software automatically creates the channel group if it does not already exist.

	Command or Action	Purpose
Step 3	**show port-channel summary** **Example:** `switch(config-router)# show port-channel summary`	(Optional) Displays information about the port channel.

Step 4	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 5	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to create a port channel:

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

See the "Compatibility Requirements" section for details on how the interface configuration changes when you delete the port channel.

Adding a Layer 2 Port to a Port Channel

You can add a Layer 2 port to a new channel group or to a channel group that already contains Layer 2 ports. The software creates the port channel associated with this channel group if the port channel does not already exist.

Note Use the **no channel-group** command to remove the port from the channel group.

Command	Purpose
no channel-group **Example:** `switch(config)# no channel-group`	Removes the port from the channel group.

Before you begin

Enable LACP if you want LACP-based port channels.

All Layer 2 member ports must run in full-duplex mode and at the same speed

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **switchport**
4. **switchport mode trunk**
5. **switchport trunk** {**allowed vlan** *vlan-id* | **native** *vlan-id*}
6. **channel-group** *channel-number* [**force**] [**mode** {**on** | **active** | **passive**}]
7. **show interface** *type slot/port*
8. **no shutdown**
9. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 1/4`
	`switch(config-if)#`
		Specifies the interface that you want to add to a channel group, and enters the interface configuration mode.

Step 3	**switchport** **Example:** `switch(config)# switchport`	Configures the interface as a Layer 2 access port.
Step 4	**switchport mode trunk** **Example:** `switch(config)# switchport mode trunk`	(Optional) Configures the interface as a Layer 2 trunk port.

Step 5	**switchport trunk** {**allowed vlan** *vlan-id* | **native** *vlan-id*}
	Example:
	`switch(config)# switchport trunk native 3`
	`switch(config-if)#`
		(Optional) Configures necessary parameters for a Layer 2 trunk port.

Step 6	**channel-group** *channel-number* [**force**] [**mode** {**on** | **active** | **passive**}] **Example:** • `switch(config-if)# channel-group 5` • `switch(config-if)# channel-group 5 force`	Configures the port in a channel group and sets the mode. The channel-number range is from 1 to 4096. This command creates the port channel associated with this channel group if the port channel does not already exist. All static port-channel interfaces are set to mode **on**. You must set all LACP-enabled port-channel interfaces to **active** or **passive**. The default mode is **on**.

	Command or Action	Purpose
		(Optional) Forces an interface with some incompatible configurations to join the channel. The forced interface must have the same speed, duplex, and flow control settings as the channel group. **Note** The **force** option fails if the port has a QoS policy mismatch with the other members of the port channel.
Step 7	**show interface** *type slot/port* **Example:** `switch# show interface port channel 5`	(Optional) Displays interface information.

Step 8	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 9	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to add a Layer 2 Ethernet interface 1/4 to channel group 5:

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

Adding a Layer 3 Port to a Port Channel

You can add a Layer 3 port to a new channel group or to a channel group that is already configured with Layer 3 ports. The software creates the port channel associated with this channel group if the port channel does not already exist.

If the Layer 3 port that you are adding has a configured IP address, the system removes that IP address before adding the port to the port channel. After you create a Layer 3 port channel, you can assign an IP address to the port-channel interface.

Note Use the **no channel-group** command to remove the port from the channel group. The port reverts to its original configuration. You must reconfigure the IP addresses for this port.

Command	Purpose
no channel-group **Example:** `switch(config)# no channel-group`	Removes the port from the channel group.

Before you begin

Enable LACP if you want LACP-based port channels.

Remove any IP addresses configured on the Layer 3 interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **no switchport**
4. **channel-group** *channel-number* [**force**] [**mode** {**on** | **active** | **passive**}]
5. **show interface** *type slot/port*
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 1/4`
	`switch(config-if)#`
		Specifies the interface that you want to add to a channel group, and enters the interface configuration mode.

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 port.

	Command or Action	Purpose
Step 4	**channel-group** *channel-number* [**force**] [**mode** {**on** | **active** | **passive**}] **Example:** • `switch(config-if)# channel-group 5` • `switch(config-if)# channel-group 5 force`	Configures the port in a channel group and sets the mode. The channel-number range is from 1 to 4096. The Cisco NX-OS software creates the port channel associated with this channel group if the port channel does not already exist. (Optional) Forces an interface with some incompatible configurations to join the channel. The forced interface must have the same speed, duplex, and flow control settings as the channel group.
Step 5	**show interface** *type slot/port* **Example:** `switch# show interface ethernet 1/4`	(Optional) Displays interface information.

Step 6	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to add a Layer 3 Ethernet interface 1/5 to channel group 6 in on mode:

```
switch# configure terminal
switch (config)# interface ethernet 1/5
switch(config-if)# switchport
switch(config-if)# channel-group 6
```

This example shows how to create a Layer 3 port-channel interface and assign the IP address:

```
switch# configure terminal
switch (config)# interface port-channel 4
switch(config-if)# ip address 192.0.2.1/8
```

Configuring the Bandwidth and Delay for Informational Purposes

The bandwidth of the port channel is determined by the number of total active links in the channel.

You configure the bandwidth and delay on port-channel interfaces for informational purposes.

SUMMARY STEPS

1. **configure terminal**

2. **interface port-channel** *channel-number*
3. **bandwidth** *value*
4. **delay** *value*
5. **exit**
6. **show interface port-channel** *channel-number*
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *channel-number*
	Example:
	`switch(config)# interface port-channel 2`
	`switch(config-if)#`
		Specifies the port-channel interface that you want to configure, and enters the interface mode.

Step 3	**bandwidth** *value*
	Example:
	`switch(config-if)# bandwidth 60000000`
	`switch(config-if)#`
		Specifies the bandwidth, which is used for informational purposes. The range is from 1 to 3,200,000,000 kbs. The default value depends on the total active interfaces in the channel group.

Step 4	**delay** *value*
	Example:
	`switch(config-if)# delay 10000`
	`switch(config-if)#`
		Specifies the throughput delay, which is used for informational purposes. The range is from 1 to 16,777,215 tens of microseconds. The default value is 10 microseconds.

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode and returns to the configuration mode.

Step 6	**show interface port-channel** *channel-number* **Example:** `switch# show interface port-channel 2`	(Optional) Displays interface information for the specified port channel.
Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure the informational parameters of the bandwidth and delay for port channel 5:

```
switch# configure terminal
switch (config)# interface port-channel 5
switch(config-if)# bandwidth 60000000
switch(config-if)# delay 10000
switch(config-if)#
```

Shutting Down and Restarting the Port-Channel Interface

You can shut down and restart the port-channel interface. When you shut down a port-channel interface, no traffic passes and the interface is administratively down.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-number*
3. **shutdown**
4. **exit**
5. **show interface port-channel** *channel-number*
6. **no shutdown**
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *channel-number*
	Example:
	`switch(config)# interface port-channel 2`
	`switch(config-if)#`
		Specifies the port-channel interface that you want to configure, and enters the interface mode.

Step 3	**shutdown**
	Example:
	`switch(config-if)# shutdown`
	`switch(config-if)#`
		Shuts down the interface. No traffic passes and the interface displays as administratively down. The default is no shutdown.
		Note
		Use the **no shutdown** command to open the interface.

	Command or Action	Purpose
		The interface displays as administratively up. If there are no operational problems, traffic passes. The default is no shutdown.

Step 4	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode and returns to the configuration mode.

Step 5	**show interface port-channel** *channel-number* **Example:** `switch(config-router)# show interface port-channel 2`	(Optional) Displays interface information for the specified port channel.

Step 6	**no shutdown**
	Example:
	`switch# configure terminal`
	`switch(config)# int e3/1`
	`switch(config-if)# no shutdown`
		(Optional) Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.

Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to bring up the interface for port channel 2:

```
switch# configure terminal
switch (config)# interface port-channel 2
switch(config-if)# no shutdown
```

Configuring a Port-Channel Description

You can configure a description for a port channel.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-number*
3. **description**
4. **exit**
5. **show interface port-channel** *channel-number*
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *channel-number*
	Example:
	`switch(config)# interface port-channel 2`
	`switch(config-if)#`
		Specifies the port-channel interface that you want to configure, and enters the interface mode.

Step 3	**description**
	Example:
	`switch(config-if)# description engineering`
	`switch(config-if)#`
		Allows you to add a description to the port-channel interface. You can use up to 80 characters in the description. By default, the description does not display; you must configure this parameter before the description displays in the output.

Step 4	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode and returns to the configuration mode.

Step 5	**show interface port-channel** *channel-number* **Example:** `switch# show interface port-channel 2`	(Optional) Displays interface information for the specified port channel.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to add a description to port channel 2:

```
switch# configure terminal
switch (config)# interface port-channel 2
switch(config-if)# description engineering
```

Configuring the Speed and Duplex Settings for a Port-Channel Interface

You can configure the speed and duplex settings for a port-channel interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-number*
3. **speed** {**10** | **100** | **1000** | **auto**}
4. **duplex** {**auto** | **full** | **half**}
5. **exit**
6. **show interface port-channel** *channel-number*
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *channel-number*
	Example:
	`switch(config)# interface port-channel 2`
	`switch(config-if)#`
		Specifies the port-channel interface that you want to configure, and enters the interface mode.

Step 3	**speed** {**10**	**100**	**1000**	**auto**}
				Example:
				`switch(config-if)# speed auto`
				`switch(config-if)#`
					Sets the speed for the port-channel interface. The default is auto for autonegotiation.

Step 4	**duplex** {**auto**	**full**	**half**}
			Example:
			`switch(config-if)# duplex auto`
			`switch(config-if)#`
				Sets the duplex for the port-channel interface. The default is auto for autonegotiation.

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits the interface mode and returns to the configuration mode.

Step 6	**show interface port-channel** *channel-number* **Example:** `switch# show interface port-channel 2`	(Optional) Displays interface information for the specified port channel.
Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set port channel 2 to 100 Mb/s:

```
switch# configure terminal
switch (config)# interface port-channel 2
switch(config-if)# speed 100
```

Configuring Load Balancing Using Port Channels

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

Note Use the **no port-channel load-balance** command to restore the default load-balancing algorithm of source-dest-mac for non-IP traffic and source-dest-ip for IP traffic.

Command	Purpose
no port-channel load-balance **Example:** `switch(config)# no port-channel load-balance`	Restores the default load-balancing algorithm.

Before you begin

Enable LACP if you want LACP-based port channels.

SUMMARY STEPS

1. **configure terminal**
2. **port-channel load-balance** *method* {**dst ip** | **dst ip-gre** | **dst ip-l4port** | **dst ip-l4port-vlan** | **dst ip-vlan** | **dst l4port** | **dst mac** | **src ip** | **src ip-gre** | **src ip-l4port** | **src ip-l4port-vlan** | **src ip-vlan** | **src l4port** | **src mac** | **src-dst ip** | **src-dst ip-gre** | **src-dst ip-l4port** [**symmetric**] | **src-dst ip-l4port-vlan** | **src-dst ip-vlan** | **src-dst l4port** | **src-dst mac**} [**fex** {*fex-range* | *all*}] [**dst inner-header**] | **src inner-header** | **src-dst inner-header**] [**rotate** *rotate*]
3. **show port-channel load-balance**
4. **show port-channel load-balance** [**forwarding-path interface port-channel** *channel-number* |**src-ip** *src-ip* |**dst-ip** *dst-ip* |**protocol** *protocol* |**gtp-teid** *gtp-teid* |**module** *module_if*]
5. **copy running-config startup-config**

DETAILED STEPS

Procedure

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

	Command or Action	Purpose
	`switch# configure terminal` `switch(config)#`	
Step 2	**port-channel load-balance** *method* {**dst ip** | **dst ip-gre** | **dst ip-l4port** | **dst ip-l4port-vlan** | **dst ip-vlan** | **dst l4port** | **dst mac** | **src ip** | **src ip-gre** | **src ip-l4port** | **src ip-l4port-vlan** | **src ip-vlan** | **src l4port** | **src mac** | **src-dst ip** | **src-dst ip-gre** | **src-dst ip-l4port** [**symmetric**] | **src-dst ip-l4port-vlan** | **src-dst ip-vlan** | **src-dst l4port** | **src-dst mac**} [**fex** {*fex-range* | *all*}] [**dst inner-header**] | **src inner-header** | **src-dst inner-header**] [**rotate** *rotate*] **Example:** • `switch(config)# port-channel load-balance src-dst mac` `switch(config)#` • `switch(config)# no port-channel load-balance src-dst mac` `switch(config)#` • `switch(config)# port-channel load-balance dst inner-header` `switch(config)#` • `switch(config)# port-channel load-balance src inner-header` `switch(config)#` • `switch(config)# port-channel load-balance src-dst inner-header` `switch(config)#`	Specifies the load-balancing algorithm for the device. The range depends on the device. The default for Layer 3 is **src-dst ip-l4port** for both IPv4 and IPv6, and the default for non-IP is **src-dst mac**. **Note** GRE inner IP headers supports source, destination and source-destination. **Note** Only the following load-balancing algorithms support symmetric hashing: • src-dst ip • src-dst ip-l4port
Step 3	**show port-channel load-balance** **Example:** `switch(config-router)# show port-channel load-balance`	(Optional) Displays the port-channel load-balancing algorithm.
Step 4	**show port-channel load-balance** [**forwarding-path interface port-channel** *channel-number* |**src-ip** *src-ip* |**dst-ip** *dst-ip* |**protocol** *protocol* |**gtp-teid** *gtp-teid* |**module** *module_if*] **Example:** `switch# show port-channel load-balance forwarding-path load-balance`	(Optional) Identifies the port in the EtherChannel interface that forwards the packet.
Step 5	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

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 group the links into a port channel. The port channel is then added to the spanning tree as a single bridge port.

To configure LACP, you must do the following:

- Enable LACP globally by using the **feature lacp** command.
- You can use different modes for different interfaces within the same LACP-enabled port channel. You can change the mode between **active** and **passive** for an interface only if it is the only interface that is designated to the specified channel group.

SUMMARY STEPS

1. **configure terminal**
2. **feature lacp**
3. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature lacp** **Example:** `switch(config)# feature lacp`	Enables LACP on the device.
Step 3	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enable LACP:

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

Configuring LACP Port-Channel Port Modes

After you enable LACP, 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 aggregation protocol, all interfaces on both sides of the link remain in the **on** channel mode.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **channel-group** *number* **mode** {**active** | **on** | **passive**}
4. **show port-channel summary**
5. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 1/4`
	`switch(config-if)#`
		Specifies the interface that you want to add to a channel group, and enters the interface configuration mode.

Step 3	**channel-group** *number* **mode** {**active**	**on**	**passive**} **Example:** `switch(config-if)# channel-group 5 mode active`	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. When you run port channels with no associated aggregation protocol, the port-channel mode is always on. The default port-channel mode is **on**.
Step 4	**show port-channel summary** **Example:** `switch(config-if)# show port-channel summary`	(Optional) Displays summary information about the port channels.
Step 5	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to set the LACP-enabled interface to the 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
```

Configuring LACP Port-Channel Minimum Links

You can configure the LACP minimum links feature. Although minimum links and maxbundles work only in LACP, you can enter the CLI commands for these features for non-LACP port channels, but these commands are nonoperational.

Note Use the **no lacp min-links** command to restore the default port-channel minimum links configuration.

Command	Purpose
no lacp min-links **Example:** `switch(config)# no lacp min-links`	Restores the default port-channel minimum links configuration.

Before you begin

Ensure that you are in the correct port-channel interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *number*
3. **lacp min-links** *number*
4. **show running-config interface port-channel** *number*

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

	Command or Action	Purpose

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 3`
	`switch(config-if)#`
		Specifies the interface to configure, and enters the interface configuration mode.

Step 3	**lacp min-links** *number* **Example:** `switch(config-if)# lacp min-links 3`	Specifies the port-channel interface to configure the number of minimum links. The range is from 1 to 16.
Step 4	**show running-config interface port-channel** *number* **Example:** `switch(config-if)# show running-config interface port-channel 3`	(Optional) Displays the port-channel minimum links configuration.

Example

This example shows how to configure the minimum number of port-channel member interfaces to be up/active for the port-channel to be up/active:

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

Configuring the LACP Port-Channel MaxBundle

You can configure the LACP maxbundle feature. Although minimum links and maxbundles work only in LACP, you can enter the CLI commands for these features for non-LACP port channels, but these commands are nonoperational.

Note Use the **no lacp max-bundle** command to restore the default port-channel max-bundle configuration.

Command	Purpose
no lacp max-bundle **Example:** `switch(config)# no lacp max-bundle`	Restores the default port-channel max-bundle configuration.

Before you begin

Ensure that you are in the correct port-channel interface.

SUMMARY STEPS

1. **configure terminal**

2. **interface port-channel** *number*
3. **lacp max-bundle** *number*
4. **show running-config interface port-channel** *number*

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 3`
	`switch(config-if)#`
		Specifies the interface to configure, and enters the interface configuration mode.

Step 3	**lacp max-bundle** *number* **Example:** `switch(config-if)# lacp max-bundle`	Specifies the port-channel interface to configure max-bundle. The default value for the port-channel max-bundle is 16. The allowed range is from 1 to 32. **Note** Even if the default value is 16, the number of active members in a port channel is the minimum of the pc_max_links_config and pc_max_active_members that is allowed in the port channel.
Step 4	**show running-config interface port-channel** *number* **Example:** `switch(config-if)# show running-config interface port-channel 3`	(Optional) Displays the port-channel max-bundle configuration.

Example

This example shows how to configure the port channel interface max-bundle:

```
switch# configure terminal
switch(config)# interface port-channel 3
switch(config-if)# lacp max-bundle 3
```

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.

Note We do not recommend changing the LACP timer rate. HA and SSO are not supported when the LACP fast rate timer is configured.

Note Configuring **lacp rate fast** is not recommended on the vPC Peer-Links. When **lacp rate fast** is configured on the vPC Peer-Link member interfaces, an alert is displayed in the syslog messages only when the LACP logging level is set to 5.

Before you begin

Ensure that you have enabled the LACP feature.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **lacp rate fast**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 1/4`
	`switch(config-if)#`
		Specifies the interface to configure and enters the interface configuration mode.

Step 3	**lacp rate fast** **Example:** `switch(config-if)# lacp rate fast`	Configures the fast rate (one second) at which LACP control packets are sent to an LACP-supported interface. To reset the timeout rate to its default, use the **no** form of the command.

Example

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

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

Before you begin

Enable LACP.

SUMMARY STEPS

1. **configure terminal**
2. **lacp system-priority** *priority*
3. **show lacp system-identifier**
4. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**lacp system-priority** *priority* **Example:** `switch(config)# lacp system-priority 40000`	Configures the system priority for use with LACP. Valid values are from 1 through 65535, and higher numbers have a lower priority. The default value is 32768. **Note** Each VDC has a different LACP system ID because the software adds the MAC address to this configured value.
Step 3	**show lacp system-identifier** **Example:** `switch(config-if)# show lacp system-identifier`	(Optional) Displays the LACP system identifier.
Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

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

When you enable LACP, you can configure each link in the LACP port channel for the port priority.

Before you begin

Enable LACP.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **lacp port-priority** *priority*
4. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 1/4`
	`switch(config-if)#`
		Specifies the interface that you want to add to a channel group, and enters the interface configuration mode.

Step 3	**lacp port-priority** *priority* **Example:** `switch(config-if)# lacp port-priority 40000`	Configures the port priority for use with LACP. Valid values are from 1 through 65535, and higher numbers have a lower priority. The default value is 32768.
Step 4	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

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
```

Configuring LACP System MAC and Role

You can configure the MAC address used by the LACP for protocol exchanges and the optional role. By default, the LACP uses the VDC MAC address. By default, the role is primary.

Use the **no lacp system-mac** command to make LACP use the default (VDC) MAC address and default role.

This procedure is supported on the Cisco Nexus 9336C-FX2, 93300YC-FX2, and 93240YC-FX2-Z switches.

Before you begin

LACP must be enabled.

SUMMARY STEPS

1. **configure terminal**
2. **lacp system-mac** *mac-address* **role** *role-value*
3. (Optional) **show lacp system-identifier**
4. **copy running-config startup-config**

DETAILED STEPS

Procedure

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

Step 2	**lacp system-mac** *mac-address* **role** *role-value*
	Example:
	`switch(config)# lacp system-mac 000a.000b.000c role primary`
	`switch(config)# lacp system-mac 000a.000b.000c role secondary`
		Specifies the MAC address to use in the LACP protocol exchanges. The role is optional. Primary is the default.

Step 3	(Optional) **show lacp system-identifier** **Example:** `switch(config)# show lacp system-identifier`	Displays the configured MAC address.

	Command or Action	Purpose
Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Example

The following example shows how to configure the role of a switch as primary.

```
Switch1# sh lacp system-identifier
32768,0-b-0-b-0-b
Switch1# sh run  | grep lacp
feature lacp
lacp system-mac 000b.000b.000b role primary
```

The following example shows how to configure the role of a switch as secondary.

```
Switch2# sh lacp system-identifier
32768,0-b-0-b-0-b
Switch2# sh run | grep lacp
feature lacp
lacp system-mac 000b.000b.000b role secondary
```

Disabling LACP Graceful Convergence

By default, LACP graceful convergence is enabled. In situations where you need to support LACP interoperability with devices where the graceful failover defaults may delay the time taken for a disabled port to be brought down or cause traffic from the peer to be lost, you can disable convergence. If the downstream access switch is not a Cisco Nexus device, disable the LACP graceful convergence option.

Note The port channel has to be in the administratively down state before the command can be run.

Before you begin

Enable LACP.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *number*
3. **shutdown**
4. **no lacp graceful-convergence**
5. **no shutdown**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 1`
	`switch(config-if)#`
		Specifies the port channel interface to configure and enters the interface configuration mode.

Step 3	**shutdown** **Example:** `switch(config-if) shutdown`	Administratively shuts down the port channel.
Step 4	**no lacp graceful-convergence** **Example:** `switch(config-if)# no lacp graceful-convergence`	Disables LACP graceful convergence on the port channel.
Step 5	**no shutdown** **Example:** `switch(config-if) no shutdown`	Brings the port channel administratively up.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to disable LACP graceful convergence on a port channel:

```
switch# configure terminal
switch (config)# interface port-channel 1
switch(config-if)# shutdown
switch(config-if)# no lacp graceful-convergence
switch(config-if)# no shutdown
```

Reenabling LACP Graceful Convergence

If the default LACP graceful convergence is once again required, you can reenable convergence.

SUMMARY STEPS

1. **configure terminal**

2. **interface port-channel** *number*
3. **shutdown**
4. **lacp graceful-convergence**
5. **no shutdown**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 1`
	`switch(config-if)#`
		Specifies the port channel interface to configure and enters the interface configuration mode.

Step 3	**shutdown** **Example:** `switch(config-if) shutdown`	Administratively shuts down the port channel.
Step 4	**lacp graceful-convergence** **Example:** `switch(config-if)# lacp graceful-convergence`	Enables LACP graceful convergence on the port channel.
Step 5	**no shutdown** **Example:** `switch(config-if) no shutdown`	Brings the port channel administratively up.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enable LACP graceful convergence on a port channel:

```
switch# configure terminal
switch (config)# interface port-channel 1
switch(config-if)# shutdown
switch(config-if)# lacp graceful-convergence
switch(config-if)# no shutdown
```

Disabling LACP Suspend Individual

LACP sets a port to the suspended state if it does not receive an LACP PDU from the peer. This process can cause some servers to fail to boot up as they require LACP to logically bring up the port.

Note You should only enter the **lacp suspend-individual** command on edge ports. The port channel has to be in the administratively down state before you can use this command.

Before you begin

Enable LACP.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *number*
3. **shutdown**
4. **no lacp suspend-individual**
5. **no shutdown**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 1`
	`switch(config-if)#`
		Specifies the port channel interface to configure and enters the interface configuration mode.

Step 3	**shutdown** **Example:** `switch(config-if) shutdown`	Administratively shuts down the port channel.
Step 4	**no lacp suspend-individual** **Example:** `switch(config-if)# no lacp suspend-individual`	Disables LACP individual port suspension behavior on the port channel.

	Command or Action	Purpose
Step 5	**no shutdown** **Example:** `switch(config-if) no shutdown`	Brings the port channel administratively up.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to disable LACP individual port suspension on a port channel:

```
switch# configure terminal
switch (config)# interface port-channel 1
switch(config-if)# shutdown
switch(config-if)# no lacp suspend-individual
switch(config-if)# no shutdown
```

Reenabling LACP Suspend Individual

You can reenable the default LACP individual port suspension.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *number*
3. **shutdown**
4. **lacp suspend-individual**
5. **no shutdown**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 1`
	`switch(config-if)#`
		Specifies the port channel interface to configure and enters the interface configuration mode.

	Command or Action	Purpose
Step 3	**shutdown** **Example:** `switch(config-if) shutdown`	Administratively shuts down the port channel.
Step 4	**lacp suspend-individual** **Example:** `switch(config-if)# lacp suspend-individual`	Enables LACP individual port suspension behavior on the port channel.
Step 5	**no shutdown** **Example:** `switch(config-if) no shutdown`	Brings the port channel administratively up.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to reenable the LACP individual port suspension on a port channel:

```
switch# configure terminal
switch (config)# interface port-channel 1
switch(config-if)# shutdown
switch(config-if)# lacp suspend-individual
switch(config-if)# no shutdown
```

Configuring Delayed LACP

The delayed LACP feature enables one port channel member, the delayed LACP port, to come up first as a member of a regular port channel before LACP PDUs are received. You configure the delayed LACP feature using the **lacp mode delay**command on a port channel followed by configuring the LACP port priority on a one member port of the port channel.

Note For vPC, you must enable the delayed LACP on both vPC switches.

Note For vPC, when the delayed LACP port is on the primary switch and the primary switch fails to boot, you need to remove the vPC configuration on the delayed LACP port-channel of the acting primary switch and flap the port-channel for a new port to be chosen as the delayed LACP port on the existing port-channel.

SUMMARY STEPS

1. **configure terminal**

2. **interface port-channel** *number*
3. **lacp mode delay**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**interface port-channel** *number*	Specifies the port channel interface to configure and enters the interface configuration mode.
Step 3	**lacp mode delay**	Enables delayed LACP. **Note** To disable delayed LACP, use the **no lacp mode delay** command. Complete the configuration of the delayed LACP by configuring the LACP port priority. See the "Configuring the LACP Port Priority" section for details. The priority of a LACP port determines the election of the delayed LACP port. The port with the lowest numerical priority is elected. When two or more ports have the same best priority, the VDC system MAC is used to determine which vPC is used. Then within a non-vPC switch or the elected vPC switch, the smallest of the ethernet port names is used. When the delayed LACP feature is configured and made effective with a port channel flap, the delayed LACP port operates as a member of a regular port channel, allowing data to be exchanged between the server and switch. After receiving the first LACP PDU, the delayed LACP port transitions from a regular port member to a LACP port member. **Note** The election of the delayed LACP port is not complete or effective until the port channel flaps on the switch or at a remote server.

Example

The following example configures delayed LACP.

```
switch# config terminal
switch(config)# interface po 1
```

```
switch(config-if)# lacp mode delay
```

```
switch# config terminal
switch(config)# interface ethernet 1/1
switch(config-if)# lacp port-priority 1
switch(config-if)# channel-group 1 mode active
```

The following example disables delayed LACP.

```
switch# config terminal
switch(config)# interface po 1
switch(config-if)# no lacp mode delay
```

Configuring Port Channel Hash Distribution

Cisco NX-OS supports the adaptive and fixed hash distribution configuration for both global and port-channel levels. This option minimizes traffic disruption by minimizing Result Bundle Hash (RBH) distribution changes when members come up or go down so that flows that are mapped to unchange RBH values continue to flow through the same links. The port-channel level configuration overrules the global configuration. The default configuration is adaptive globally, and there is no configuration for each port channel, so there is no change during an ISSU. No ports are flapped when the command is applied, and the configuration takes effect at the next member link change event. Both modes work with RBH module or non-module schemes.

During an ISSD to a lower version that does not support this feature, you must disable this feature if the fixed mode command is being used globally or if there is a port-channel level configuration.

Configuring Port Channel Hash Distribution at the Global Level

SUMMARY STEPS

1. **configure terminal**
2. **no port-channel hash-distribution** {**adaptive** | **fixed**}
3. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**no port-channel hash-distribution** {**adaptive**	**fixed**} **Example:**	Specifies the port-channel hash distribution at the global level. The default is adaptive mode.

	Command or Action	Purpose
	`switch(config)# port-channel hash-distribution adaptive` `switch(config)#`	The command does not take effect until the next member link event (link down/up/no shutdown/shutdown). (`Do you still want to continue(y/n)? [yes]`)
Step 3	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure hash distribution at the global level:

```
switch# configure terminal
switch(config)# no port-channel hash-distribution fixed
```

Configuring Port Channel Hash Distribution at the Port Channel Level

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** {*channel-number* | *range*}
3. **no port-channel port hash-distribution** {**adaptive** | **fixed**}
4. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** {*channel-number*	*range*}
		Example:
		`switch# interface port-channel 4`
		`switch(config-if)#`
			Specifies the interface to configure, and enters the interface configuration mode.

Step 3	**no port-channel port hash-distribution** {**adaptive**	**fixed**}
		Example:
		`switch(config-if)# port-channel port hash-distribution adaptive`
		`switch(config-if)`
			Specifies the port-channel hash distribution at the port channel level.
			There is no default.
			The command does not take effect until the next member link event (link down/up/no shutdown/shutdown). (`Do you still want to continue(y/n)? [yes]`)

	Command or Action	Purpose
Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure hash distribution as a global-level command:

```
switch# configure terminal
switch(config)# no port-channel hash-distribution fixed
```

Enabling ECMP Resilient Hashing

Resilient ECMP ensures minimal impact to the existing flows when members are deleted from an ECMP group. This is achieved by replicating the existing members in a round-robin fashion at the indices that were previously occupied by the deleted members.

SUMMARY STEPS

1. **configure terminal**
2. **hardware profile ecmp resilient**
3. **copy running-config startup-config**
4. **reload**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**hardware profile ecmp resilient** **Example:** `switch(config)# hardware profile ecmp resilient`	Enables ECMP resilient hashing and displays the following: **Warning: The command will take effect after next reload.** **Note** This command is not supported on Cisco Nexus 9808 platform switches.
Step 3	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

	Command or Action	Purpose
Step 4	**reload** **Example:** `switch(config)# reload`	Reboots the switch.

Disabling ECMP Resilient Hashing

Before you begin

ECMP resilient hashing is enabled.

SUMMARY STEPS

1. **configure terminal**
2. **no hardware profile ecmp resilient**
3. **copy running-config startup-config**
4. **reload**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**no hardware profile ecmp resilient** **Example:** `switch(config)# no hardware profile ecmp resilient`	Disables ECMP resilient hashing and displays the following: **Warning: The command will take effect after next reload.**
Step 3	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.
Step 4	**reload** **Example:** `switch(config)# reload`	Reboots the switch.

Configuring ECMP Load Balancing

To configure the ECMP load-sharing algorithm, use the following command in global configuration mode:

Before you begin

SUMMARY STEPS

1. **ip load-sharing address** {**destination port destination** | **source-destination** [**port source-destination** | **gre** | **gtpu** | **ipv6-flowlabel** | **ttl** | **udf offset** *offset* **length** *length* | **symmetricinner** *allgreheader*]} [**universal-id** *seed*] [**rotate** *rotate*] [**concatenation**]
2. (Optional) **show ip load-sharing**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**ip load-sharing address** {**destination port destination** | **source-destination** [**port source-destination** | **gre** | **gtpu** | **ipv6-flowlabel** | **ttl** | **udf offset** *offset* **length** *length* | **symmetricinner** *allgreheader*]} [**universal-id** *seed*] [**rotate** *rotate*] [**concatenation**]
	Example:
	`ip load-sharing address source-destination`
	Example:
	`switch(config)# ip load-sharing address source-destination ipv6-flowlabel`
	Example:
	`switch(config)# ip load-sharing address source-destination ttl`
	Example:
	`switch(config)# ip load-sharing address source-destination udf offset 8 length 8`
	Example:
	`switch(config)# [no] ip load-sharing address source-destination port source-destination symmetric`
	Example:
	`switch(config)# ip load-sharing address source-destination port source-destination inner [all|greheader]`
		Configures the ECMP load-sharing algorithm for data traffic.
		• The **gre** option specifies the source-destination value for the Generic Routing Encapsulation (GRE) key.
		• The **gtpu** option specifies the GPRS Tunneling Protocol (GTP) tunnel endpoint identifier (TEID) value for the port source-destination.
		• The **ipv6-flowlabel** option includes the IPv6 flow label for computing ECMP hashing. It ensures that traffic flows are distributed on all links based on different flow label values. Enabling or disabling this option also enables or disables it for port-channel load-balancing if Layer 4 parameters are enabled using the **port-channel load-balance** command. Only the following devices support this option:
		  • Cisco Nexus 9364C and 9300-EX/FX/FX2 platform switches
		  • Cisco Nexus 9500 platform switches with X9700-EX/FX line cards and FM-E2 fabric modules in all routing modes
		  • Cisco Nexus 9500 platform switches with X9700-EX/FX line cards and FM-E fabric modules in non-hierarchical routing modes where IPv6 routes are programmed in the line card
		  • Beginning with Cisco NX-OS Release 9.3(5), Cisco Nexus N9K-C9316D-GX, N9K-C93600CD-GX, N9K-C9364C-GX switches support this option.
		• The **ttl** option includes time-to-live information for computing ECMP hashing. It ensures that traffic flows are distributed on all links based on different TTL

	Command or Action	Purpose
		values. For IPv4 flows, it is based on ttl values. For IPv6 flows, it is based on hop limit. Enabling or disabling this option also enables or disables it for port-channel load-balancing if Layer 4 parameters are enabled using the **port-channel load-balance** command. Only Cisco Nexus 9364C and 9300-EX/FX/FX2 platform switches support this option. Beginning with Cisco NX-OS Release 9.3(5), Cisco Nexus N9K-C9316D-GX, N9K-C93600CD-GX, N9K-C9364C-GX switches support this option. • The **udf** option includes the user-defined field for computing ECMP hashing. You can configure the offset base and the length of the UDF field (in bits). The range for the offset base is from 0 to 127 bytes. The range for the length of the UDF field is from 1 to 32 bits. Enabling or disabling this option also enables or disables it for port-channel load-balancing if Layer 4 parameters are enabled using the **port-channel load-balance** command. Only Cisco Nexus 9364C and 9300-EX/FX/FX2 platform switches support this option. Beginning with Cisco NX-OS Release 9.3(5), Cisco Nexus N9K-C9316D-GX, N9K-C93600CD-GX, N9K-C9364C-GX switches support this option. • The **symmetric** option enables symmetric hashing globally. To disable ECMP symmetric hashing, use the **no** keyword in the command. You must execute this command in global configuration mode. **Note** Ensure that the configured **universal-id** seed value is consistent across the nodes in the path of ECMP symmetric hashing for symmetric hashing to work effectively. • The **inner** option enables inner header based hashing for GRE traffic globally. To disable inner header based hashing, use the **no** keyword in the command. You must execute this command in global configuration mode. • **all** : Configuring this option for GRE encapsulated packets starts using inner headers to hash onto a path in ECMP, which may impact other encapsulation types as well. This is supported on Cisco Nexus 9364C and 9300-EX/FX/FX2 platform switches; and Cisco Nexus 9500 platform switches with X9700-EX/FX line cards.

	Command or Action	Purpose
		• **greheader** : Configuring this option only for GRE encapsulated packets, starts using inner headers to hash onto a path in ECMP. This is supported on Cisco Nexus 9364C and 9300-FX/FX2 platform switches; and Cisco Nexus 9500 platform switches with X9700-FX line cards. The following options are available for all IP load sharing configurations: • The **universal-id** option sets the random seed for the hash algorithm and shifts the flow from one link to another. You do not need to configure the universal ID. Cisco NX-OS chooses the universal ID if you do not configure it. The *universal-id* range is from 1 to 4294967295. • The **rotate** option causes the hash algorithm to rotate the link picking selection so that it does not continually choose the same link across all nodes in the network. It does so by influencing the bit pattern for the hash algorithm. This option shifts the flow from one link to another and load balances the already load-balanced (polarized) traffic from the first ECMP level across multiple links. If you specify a *rotate* value, the 64-bit stream is interpreted starting from that bit position in a cyclic rotation. The *rotate* range is from 1 to 63, and the default is 32. **Note** With multi-tier Layer 3 topology, polarization is possible. To avoid polarization, use a different rotate bit at each tier of the topology. **Note** To configure a rotation value for port channels, use the **port-channel load-balance src-dst ip-l4port rotate** *rotate* command. • The **concatenation** option ties together the hash tag values for ECMP and the hash tag values for port channels in order to use a stronger 64-bit hash. If you do not use this option, you can control ECMP load-balancing and port-channel load-balancing independently. The default is disabled.

	Command or Action	Purpose
Step 2	(Optional) **show ip load-sharing** **Example:** `switch(config)# show ip load-sharing address source-destination`	Displays the ECMP load-sharing algorithm for data traffic.

Verifying the ECMP Resilient Hashing Configuration

To display ECMP Resilient Hashing configuration information, perform one of the following tasks:

Command	Purpose

`switch(config)# show running-config | grep "hardware profile ecmp resilient`
`hardware profile ecmp resilient`
`switch(config)#`
	Displays the enabled status.

`switch(config)# show running-config | grep "hardware profile ecmp resilient` `switch(config)#`	Displays the disabled status.

Verifying the Port-Channel Configuration

To display port-channel configuration information, perform one of the following tasks:

Command	Purpose
show interface port-channel *channel-number*	Displays the status of a port-channel interface.
show feature	Displays enabled features.
load- interval {**interval** *seconds* {**1** | **2** | **3**}}	Sets three different sampling intervals to bit-rate and packet-rate statistics.
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 load-balance	Displays the type of load balancing in use for port channels.
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.

Command	Purpose
show lacp {**counters** [**interface port-channel** *channel-number*] | [**interface** *type/slot*] | **neighbor** [**interface port-channel** *channel-number*] | **port-channel** [**interface port-channel** *channel-number*] | **system-identifier**]]}	Displays information about LACP.
show running-config interface port-channel *channel-number*	Displays information about the running configuration of the port-channel.

Monitoring the Port-Channel Interface Configuration

Use the following commands to display port-channel interface configuration information.

Command	Purpose
clear counters interface port-channel *channel-number*	Clears the counters.
clear lacp counters [**interface port-channel** *channel-number*]	Clears the LACP counters.
load- interval {**interval** *seconds* {**1** | **2** | **3**}}	Sets three different sampling intervals to bit-rate and packet-rate statistics.
show interface counters [**module** *module*]	Displays input and output octets unicast packets, multicast packets, and broadcast packets.
show interface counters detailed [**all**]	Displays input packets, bytes, and multicast and output packets and bytes.
show interface counters errors [**module** *module*]	Displays information about the number of error packets.
show lacp counters	Displays statistics for LACP.

Example Configurations for Port Channels

This example shows how to create an LACP port channel and add two Layer 2 interfaces to that port channel:

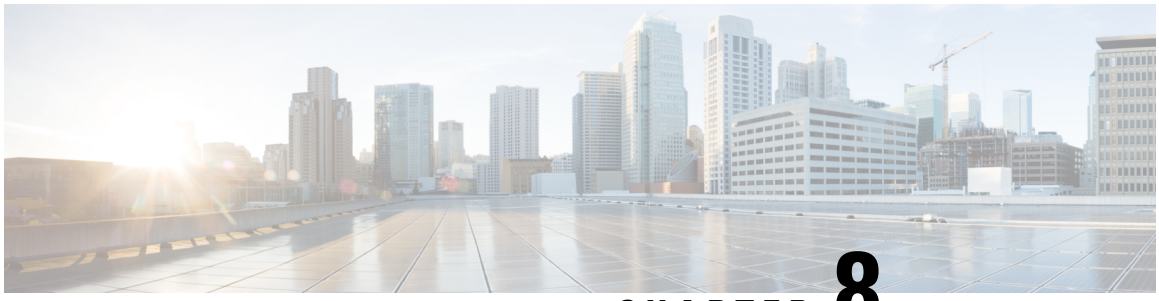
```
switch# configure terminal
switch (config)# feature lacp
switch (config)# interface port-channel 5
switch (config-if)# interface ethernet 1/4
switch(config-if)# switchport
switch(config-if)# channel-group 5 mode active
switch(config-if)# lacp port priority 40000
switch(config-if)# interface ethernet 1/7
switch(config-if)# switchport
switch(config-if)# channel-group 5 mode
```

This example shows how to add two Layer 3 interfaces to a channel group. The Cisco NX-OS software automatically creates the port channel:

```
switch# configure terminal
switch (config)# interface ethernet 1/5
switch(config-if)# no switchport
switch(config-if)# no ip address
switch(config-if)# channel-group 6 mode active
switch (config)# interface ethernet 2/5
switch(config-if)# no switchport
switch(config-if)# no ip address
switch(config-if)# channel-group 6 mode active
switch (config)# interface port-channel 6
switch(config-if)# ip address 192.0.2.1/8
```

Related Documents

Related Topic	Document Title
System management	*Cisco Nexus 9000 Series NX-OS System Management Configuration Guide*
High availability	*Cisco Nexus 9000 Series NX-OS High Availability and Redundancy Guide*
Licensing	*Cisco NX-OS Licensing Guide*



CHAPTER 8

Configuring vPCs

Information About vPCs

vPC Overview

A virtual port channel (vPC) allows links that are physically connected to two Cisco Nexus 9000 Series devices to appear as a single port channel by a third device (see figure). The third device can be a switch, server, or any other networking device that supports port channels. A vPC can provide Layer 2 multipathing, which allows you to create redundancy and increase the bisectional bandwidth by enabling multiple parallel paths between nodes and allowing load balancing traffic.

- Allows a single device to use a port channel 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 device fails
- Provides link-level resiliency
- Assures high availability

The virtual port channel (vPC) is a technology that allows a single downstream device to connect to two upstream devices as though they were one logical device.

- Layer 2 port channel support
- Link Aggregation Control Protocol (LACP) optional
- Enables redundancy and load balancing

vPC supports trunk mode port channels with or without LACP, and improves network stability and convergence.

vPC Protocol Details and Recommendations

You can use only Layer 2 port channels in the vPC. You configure the port channels by using one of the following:

- No protocol
- Link Aggregation Control Protocol (LACP)

When you configure the port channels in a vPC—including the vPC Peer-Link channel—without using LACP, each device can have up to 32 active links in a single port channel. When using LACP, each device can have 32 active links and eight standby links.

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

The system automatically takes a checkpoint prior to disabling the feature, and you can roll back to this checkpoint.

After you enable the vPC functionality, you create the peer-keepalive link, which sends heartbeat messages between the two vPC peer devices.

To ensure that you have the correct hardware to enable and run a vPC, enter the **show hardware feature-capability** command. If you see an X across from the vPC in your command output, your hardware cannot enable the vPC feature.

Note Devices attached to a vPC domain using port channels should be connected to both of vPC peers.

Figure 11: vPC Architecture

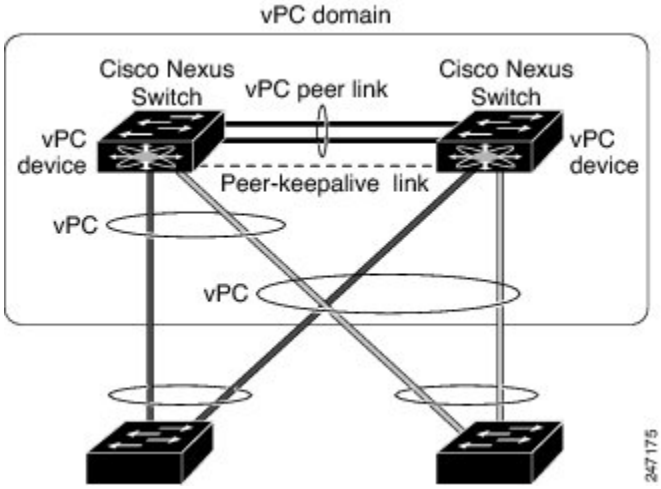
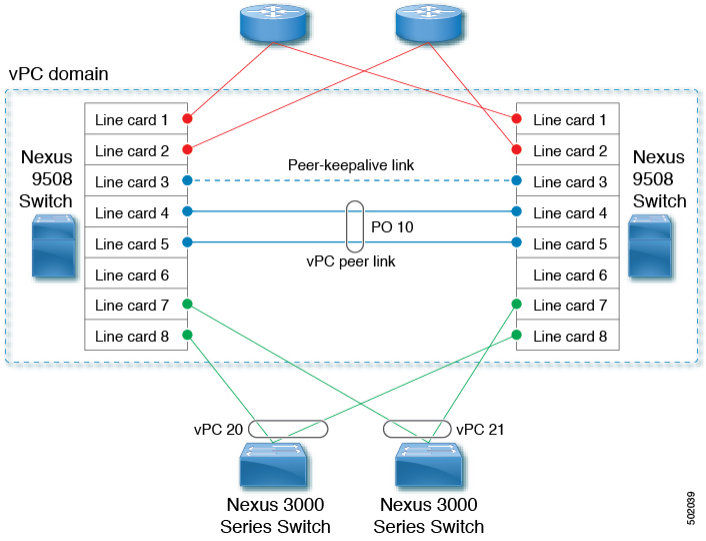


Figure 12: vPC Interfaces



Example: vPC Peer-Link Creation

You can create a vPC Peer-Link by configuring a port channel on one Cisco Nexus 9000 Series chassis by using two or more Ethernet ports higher speed than 1-Gigabit Ethernet.

We recommend that you configure the vPC Peer-Link Layer 2 port channels as trunks. On another Cisco Nexus 9000 Series chassis, you configure another port channel again using two or more Ethernet ports with speed higher than 1-Gigabit in the dedicated port mode.

Connecting these two port channels creates a vPC Peer-Link in which the two linked Cisco Nexus devices appear as one device to a third device.

Incorrect Hardware or Module Usage

If you are not using the correct module, the system displays an error message.

Once you configure this feature and if the primary vPC peer device fails, the system automatically suspends all the vPC links on the primary vPC peer device.

Track Object Recommendation

You can create a track object and apply that object to all links on the primary vPC peer device that connect to the core and to the vPC Peer-Link.

If you must configure all the vPC Peer-Links and core-facing interfaces on a single module, you should configure a track object.

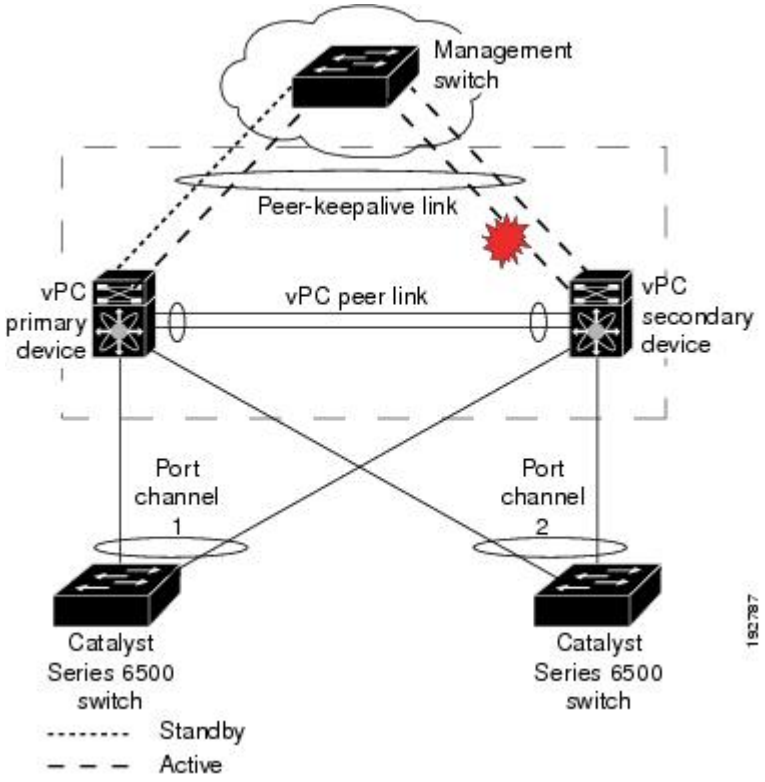
vPC Terminology

The terminology used in vPCs is as follows:

- vPC—The combined port channel between the vPC peer devices and the downstream device.
- vPC peer device—One of a pair of devices that are connected with the special port channel known as the vPC Peer-Link.
- vPC Peer-Link—The link used to synchronize state between the vPC peer devices. This link must use a 10-Gigabit Ethernet interface at a minimum. Higher-bandwidth interfaces (such as 25-Gigabit Ethernet, 40-Gigabit Ethernet, 100-Gigabit Ethernet, and so on) may also be used.
- vPC member port—An interface that belongs to a vPC.
- Host vPC port—A Fabric Extender host interface that belongs to a vPC.
- vPC domain—This domain 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.
- vPC peer-keepalive link—The peer-keepalive link monitors the vitality of a vPC peer Cisco Nexus 9000 Series device. The peer-keepalive link sends configurable, periodic keepalive messages between vPC peer devices.

  We recommend that you associate a peer-keepalive link to a separate virtual routing and forwarding (VRF) instance that is mapped to a Layer 3 interface in each vPC peer device. If you do not configure a separate VRF, the system uses the management VRF by default. However, if you use the management interfaces for the peer-keepalive link, you must put a management switch connected to both the active and standby management ports on each vPC peer device (see figure).

Figure 13: Separate Switch Required to Connect Management Ports for vPC Peer-Keepalive Link



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 a vPC.

- Dual-active—Both vPC peers act as primary. This situation occurs when the peer-keepalive and vPC Peer-Link go down while both peers are still active. In this case, the secondary vPC assumes that the primary vPC is inactive and acts as the primary vPC.
- Recovery—When the peer-keepalive and the vPC Peer-Link come up, one switch becomes the secondary vPC. On the switch that becomes the secondary vPC, the vPC links go down and come back up.

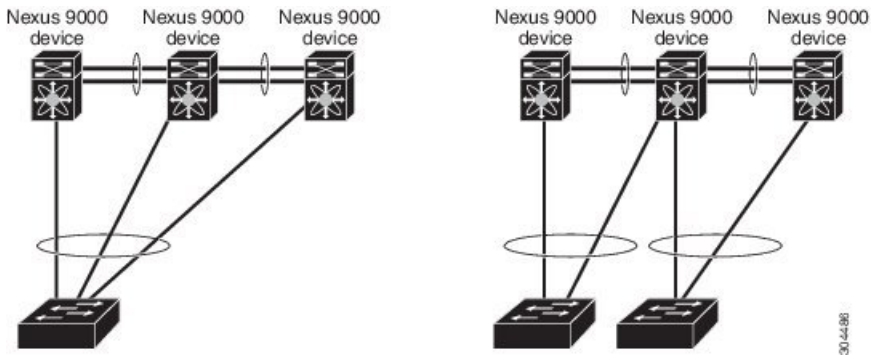
vPC Peer-Link Overview

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

Invalid vPC Peer Configurations

See the following figure for invalid vPC peer configurations.

Figure 14: vPC Peer Configurations That Are Not Allowed



Configuring vPC Peer-Link and Redundancy

To make a valid configuration, you first configure a port channel on each device and then configure the vPC domain. You assign the port channel on each device as a vPC Peer-Link, using the same vPC domain ID. For redundancy, we recommend that you should configure at least two of the dedicated ports into the port channel because if one of the interfaces in the vPC Peer-Link fails, the device automatically falls back to use another interface in the vPC Peer-Link.

Note We recommend that you configure the Layer 2 port channels in trunk mode.

Compatibility and Configuration Consistency

Many operational parameters and configuration parameters must be the same in each device connected by a vPC Peer-Link (see the Compatibility Parameters for vPC Interfaces section). Because each device is completely independent on the management plane, you must ensure that the devices are compatible on the critical parameters. vPC peer devices have separate control planes. After configuring the vPC Peer-Link, you should display the configuration on each vPC peer device to ensure that the configurations are compatible.

Note You must ensure that the two devices connected by the vPC Peer-Link have certain identical operational and configuration parameters. For more information on required configuration consistency, see the Compatibility Parameters for vPC Interfaces section.

Primary and Secondary Device Roles

When you configure the vPC Peer-Link, the vPC peer devices negotiate that one of the connected devices is the primary device and the other connected device is the secondary device (see the “Configuring vPCs” section). By default, the Cisco NX-OS software uses the lowest MAC address to elect the primary device. However, if the role priority is set, then the device with the lowest priority will be elected as the primary device. The software takes different actions on each device—that is, the primary and secondary—only in certain failover conditions. If the primary device fails, the secondary device becomes the new primary device when the system recovers, and the previously primary device is now the secondary device.

You can also configure which of the vPC devices is the primary device. Changing the priority of the vPC peer devices can cause the interfaces in your network to go up and down. If you want to configure the role priority

again to make one vPC device the primary device, configure the role priority on both the primary vPC device with a lower priority value and the secondary vPC device with the higher value. Then, shut down the port channel that is the vPC Peer-Link on both devices by entering the **shutdown** command, and finally reenable the port channel on both devices by entering the **no shutdown** command.

Note We recommend that you use two different modules for redundancy on each vPC peer device on each vPC Peer-Link.

Traffic Flow and Load Balancing

The software keeps all traffic that forwards across the vPC peer devices as local traffic. A packet that ingresses the port channel uses one of the local links rather than moving across the vPC Peer-Link. Unknown unicast, multicast, and broadcast traffic (including STP BPDUs) are flooded across the vPC Peer-Link. The software keeps the multicast forwarding state synchronized on both of the vPC peer devices.

You can configure any of the standard load-balancing schemes on both the vPC Peer-Link devices and the downstream device (see the *Configuring Port Channels* chapter for information about load balancing).

Configuration and MAC Address Synchronization

Configuration information flows across the vPC Peer-Links using the Cisco Fabric Services over Ethernet (CFSoE) protocol. (See the CFSoE, on page 257 section for more information about CFSoE.)

All MAC addresses for those VLANs configured on both devices are synchronized between vPC peer devices. The software uses CFSoE for this synchronization. (See the CFSoE, on page 257 section for information about CFSoE.)

vPC Peer-Link Failure and Peer-Keepalive

If the vPC Peer-Link fails, the software checks the status of the remote vPC peer device using the peer-keepalive link, which is a link between vPC peer devices that ensures that both devices are up. If the vPC peer device is up, the secondary vPC device disables all vPC ports on its device, to prevent loops and disappearing or flooding traffic. The data then forwards down the remaining active links of the port channel.

The software learns of a vPC peer device 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 devices. 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 device. The keepalive messages are used only when all the links in the vPC Peer-Link fail. See the “Peer-Keepalive Link and Messages” section for information about the keepalive message.

Features That You Must Manually Configure on the Primary and Secondary Devices

You must manually configure the following features to conform to the primary/secondary mapping of each of the vPC peer devices.

STP Root Configuration

STP root—Configure the primary vPC peer device as the STP primary root device and configure the vPC secondary device to be the STP secondary root device. See the "vPC Peer-Links and STP" section for more information about vPCs and STP.

- We recommend that you configure the vPC Peer-Link interfaces as STP network ports so that Bridge Assurance is enabled on all vPC Peer-Links.
- We recommend that you configure Rapid per VLAN Spanning Tree plus (PVST+) so that the primary device is the root for all VLANs and configure Multiple Spanning Tree (MST) so that the primary device is the root for all instances.

Layer 3 VLAN Network Interface Configuration

Layer 3 VLAN network interface—Configure Layer 3 connectivity from each vPC peer device by configuring a VLAN network interface for the same VLAN from both devices.

HSRP Active Configuration

HSRP active—If you want to use Hot Standby Router Protocol (HSRP) and VLAN interfaces on the vPC peer devices, configure the primary vPC peer device with the HSRP active highest priority. Configure the secondary device to be the HSRP standby and ensure that you have VLAN interfaces on each vPC device that are in the same administrative and operational mode. (See the "vPC Peer-Links and Routing" section for more information on vPC and HSRP.)

UDLD Configuration Recommendations

While you configure Unidirectional Link Detection (UDLD), note the following recommendations:

- If LACP is used as port-channel aggregation protocol, UDLD is not required in a vPC domain.
- If LACP is not used as the port-channel aggregation protocol (static port-channel), use UDLD in normal mode on vPC member ports.
- If STP is used without Bridge Assurance and if LACP is not used, use UDLD in normal mode on vPC orphan ports.

Peer-Keepalive Link and 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.

Note We recommend that you associate the vPC peer-keepalive link to a separate VRF mapped to a Layer 3 interface in each vPC peer device. If you do not configure a separate VRF, the system uses the management VRF and management ports by default. Do not use the vPC Peer-Link itself to send and receive vPC peer-keepalive messages.

Failure Detection and Keepalive Timers

If one of the vPC peer devices fails, the vPC peer device on the other side of the vPC Peer-Link senses the failure by not receiving any peer-keepalive messages. The default interval time for the vPC peer-keepalive message is 1 second, and you can configure the interval between 400 milliseconds and 10 seconds.

You can configure a hold-timeout value with a range of 3 to 10 seconds; the default hold-timeout value is 3 seconds. This timer starts when the vPC Peer-Link goes down. During this hold-timeout period, the secondary vPC peer device ignores vPC peer-keepalive messages, which ensures that network convergence occurs before a vPC action takes place. The purpose of the hold-timeout period is to prevent false-positive cases.

You can also configure a timeout value with a range of 3 to 20 seconds; the default timeout value is 5 seconds. This timer starts at the end of the hold-timeout interval. During the timeout period, the secondary vPC peer device checks for vPC peer-keepalive hello messages from the primary vPC peer device. If the secondary vPC peer device receives a single hello message, that device disables all vPC interfaces on the secondary vPC peer device.

Hold-Timeout vs. Timeout Parameters

The difference between the hold-timeout and the timeout parameters is as follows:

- During the hold-timeout, the vPC secondary device does not take any action based on any keepalive messages received, which prevents the system taking action when the keepalive might be received just temporarily, such as if a supervisor fails a few seconds after the vPC Peer-Link goes down.
- During the timeout, the vPC secondary device takes action to become the vPC primary device if no keepalive message is received by the end of the configured interval.

See the “Configuring vPC Keepalive Link and Messages” section for information about configuring the timer for the keepalive messages.

Note Ensure that both the source and destination IP addresses used for the peer-keepalive messages are unique in your network and these IP addresses are reachable from the VRF associated with the vPC peer-keepalive link.

Peer-keepalive IP addresses must be global unicast addresses. Link-local addresses are not supported.

Configuring Trusted Ports for Peer-Keepalive

Use the command-line interface (CLI) to configure the interfaces you are using the vPC peer-keepalive messages as trusted ports. Leave the precedence at the default (6) or configure it higher.

vPC Domain

You can use the vPC domain ID to identify the vPC Peer-Links and the ports that are connected to the vPC downstream devices.

The vPC domain is also a configuration mode that you use to configure the keepalive messages and other vPC Peer-Link parameters rather than accept the default values. See the “Configuring vPCs” section for more information about configuring these parameters.

vPC Domain Creation and Peer-Link Configuration

To create a vPC domain, you must first create a vPC domain ID on each vPC peer device using a number from 1 to 1000. You can have only one vPC domain per vPC peer.

You must explicitly configure the port channel that you want to act as the vPC Peer-Link on each device. You associate the port channel that you made a vPC Peer-Link on each device with the same vPC domain ID to form a single vPC domain. Within this domain, the system provides a loop-free topology and Layer 2 multipathing.

You can only configure these port channels and vPC Peer-Links statically. You can configure the port channels and vPC Peer-Links either using LACP or no protocol. We recommend that you use LACP with the interfaces in active mode to configure port channels in each vPC, which ensures an optimized, graceful recovery in a port-channel failover scenario and provides configuration checks against configuration mismatches among the port channels themselves.

vPC System MAC Address Assignment

The vPC peer devices 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 devices 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 Layer 2 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.

See the “vPC and Orphan Ports” section for more information about displaying the vPC MAC table.

vPC Domain System Priority

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

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

vPC Topology

In both topologies, port channels P020 and P0200 must be configured identically on the peer switches and configuration synchronization is used to synchronize the configurations of the vPC switches.

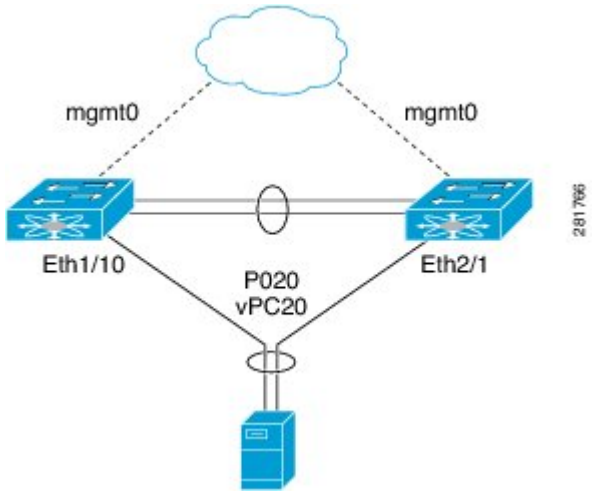
Summary

This document describes two common vPC topologies: a basic configuration with directly connected Cisco Nexus 9000 Series devices and a configuration involving Fabric Extenders (FEXs) for host vPC.

Workflow

1. The first topology shows a basic configuration in which the Cisco Nexus 9000 Series device ports are directly connected to another switch or host and are configured as part of a port channel that becomes part of a vPC.

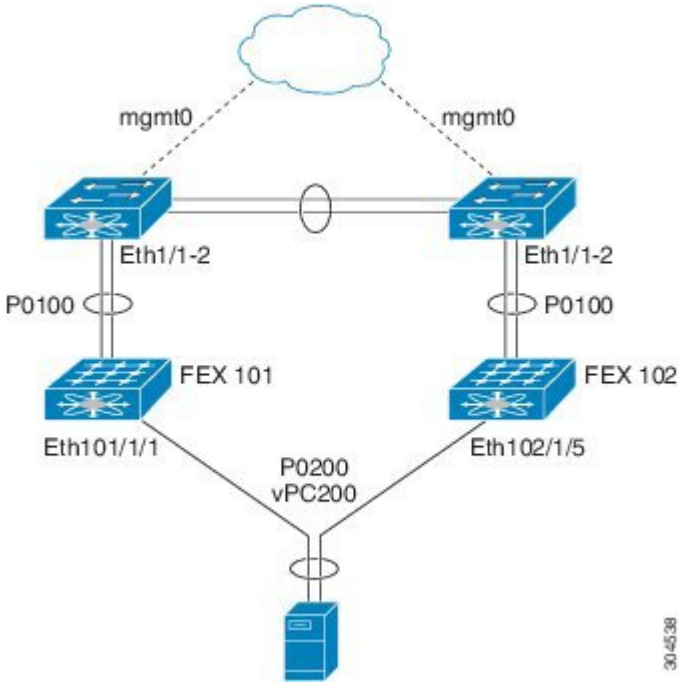
Figure 15: Switch vPC Topology



In this configuration, vPC 20 is configured on port channel 20, which has Eth1/10 on the first device and Eth2/1 on the second as member ports.

2. The second topology illustrates how to configure a vPC from the peer devices through Fabric Extenders (FEXs).

Figure 16: FEX Straight-Through Topology (Host vPC)



In this FEX straight-through topology, each FEX is single-homed with a Cisco Nexus 9000 Series device. The host interfaces on this FEX are configured as port channels, and those port channels are configured as vPCs. For example, Eth101/1/1 and Eth102/1/5 are configured as members of PO200, and PO200 is configured for vPC 200.

What's next

See the Cisco Nexus 2000 Series NX-OS Fabric Extender Configuration Guide for Cisco Nexus 9000 Series Switches for more information about configuring FEX ports.

Compatibility Parameters for vPC Interfaces

Many configuration and operational parameters must be identical on all interfaces in the vPC. We recommend that you configure the Layer 2 port channels that you use for the vPC Peer-Link in trunk mode.

After you enable the vPC feature and configure the vPC Peer-Link on both vPC peer devices, Cisco Fabric Services (CFS) messages provide a copy of the configuration on the local vPC peer device configuration to the remote vPC peer device. The system then determines whether any of the crucial configuration parameters differ on the two devices. (See the "vPC and Orphan Ports" section for more information about CFS.)

The vPC Peer-Link is a core component of vPC functionality, requiring consistent configuration across both peer devices.

- Layer 2 port channels for vPC Peer-Link must be configured in trunk mode.
- Compatibility parameters must be identical across all interfaces in the vPC.

For example, the compatibility check process differs for vPCs compared to regular port channels.

Configuration and Guidelines

After enabling the vPC feature and configuring the vPC Peer-Link, Cisco Fabric Services (CFS) ensures configuration consistency between the local and remote vPC peer devices.

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

Note The port channel compatibility parameters must be the same for all the port channel members on the physical switch. You cannot configure shared interfaces to be part of a vPC.

See the "Configuring Port Channels" chapter for more details about regular port channels.

Configuration Parameters That Must Be Identical

The configuration parameters in this section must be configured identically on both devices of the vPC Peer-Link; otherwise, the vPC moves fully or partially into a suspended mode.

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

Note 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 devices automatically check for compatibility for some 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 (port-channel mode can, however, be active/passive on each side of the vPC peer)
- 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
- Enable/disable state per VLAN
- STP global settings:
  - Bridge Assurance setting
  - Port type setting
  - Loop Guard settings
- STP interface settings:
  - Port type setting
  - Loop Guard
  - Root Guard
- Maximum Transmission Unit (MTU)

If any of these parameters are not enabled or defined on either device, 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.

In the output of **show vpc** or **show vpc brief** command, after every 50th configured vPC port-channel the following message will be displayed:

```
Please check "show vpc consistency-parameters vpc <vpc-num>" for the
consistency reason of down vpc and for type-2 consistency reasons for any
vpc.
```

Configuration Parameters That Should Be Identical

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

- MAC aging timers
- Static MAC entries
- VLAN interface—Each device 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 device of the vPC Peer-Link do not pass traffic using the vPC or vPC Peer-Link. You must create all VLANs on both the primary and secondary vPC devices, or the VLAN will be suspended.
- All ACL configurations and parameters
- Quality of Service (QoS) configuration and parameters
- STP interface settings:
  - BPDU Filter
  - BPDU Guard
  - Cost
  - Link type
  - Priority
  - VLANs (Rapid PVST+)
- Port security
- Cisco Trusted Security (CTS)
- Dynamic Host Configuration Protocol (DHCP) snooping
- Network Access Control (NAC)
- Dynamic ARP Inspection (DAI)
- IP source guard (IPSG)
- Internet Group Management Protocol (IGMP) snooping

- Hot Standby Routing Protocol (HSRP)
- Protocol Independent Multicast (PIM)
- All routing protocol configurations

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

Consequences of Parameter Mismatches

You can configure the graceful consistency check feature, which suspends only the links on the secondary peer device when a mismatch is introduced in a working vPC. This feature is configurable only in the CLI and is enabled by default.

Consistency Check Behavior

The graceful consistency-check command is configured by default.

As part of the consistency check of all parameters from the list of parameters that must be identical, the system checks the consistency of all VLANs.

The vPC remains operational, and only the inconsistent VLANs are brought down. This per-VLAN consistency check feature cannot be disabled and does not apply to Multiple Spanning Tree (MST) VLANs.

Deleting the vPC port-channel on the switch results in the suspension of the allowed VLANs on the corresponding vPC port-channel on the peer switch, regardless of the vPC role.

vPC Number

Once you have created the vPC domain ID and the vPC Peer-Link, you create port channels to attach the downstream device to each vPC peer device. That is, you create one port channel to the downstream device from the primary vPC peer device and you create another port channel to the downstream device from the secondary peer device.

Note We recommend that you configure the ports on the downstream devices that connect to a host or a network device that is not functioning as a switch or a bridge as STP edge ports.

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

Note The vPC number that you assign to the port channel that connects to the downstream device from the vPC peer device must be identical on both vPC peer devices.

Hitless vPC Role Change

A virtual port channel (vPC) allows links that are physically connected to two different Cisco Nexus 9000 Series devices to appear as a single port channel. The vPC role change feature enables you switch vPC roles between vPC peers without impacting traffic flow. The vPC role switching is done based on the role priority value of the device under the vPC domain. A vPC peer device with lower role priority is selected as the primary vPC device during the vPC Role switch. You can use the vpc role preempt command to switch vPC role between peers.

For information about how to configure Hitless vPC Role Change, see Configuring Hitless vPC Role Change, on page 292.

Moving Other Port Channels into a vPC

Note You must attach a downstream device using a port channel to both vPC peer devices.

To connect to the downstream device, you create a port channel to the downstream device from the primary vPC peer device and you create another port channel to the downstream device from the secondary peer device. On each vPC peer device, you assign a vPC number to the port channel that connects to the downstream device. You will experience minimal traffic disruption when you are creating vPCs.

vPC Object Tracking

Note We recommend that you configure the vPC Peer-Links on dedicated ports of different modules on Cisco Nexus 9500 devices. This is recommended to reduce the possibility of a failure. For the best resiliency scenario, use at least two modules.

vPC object tracking is used to prevent traffic black-holing in case of failure of a module where both vPC Peer-Link and uplinks to the core resides. By tracking interface feature can suspend vPC on affected switch and prevent traffic black-holing.

If you must configure all the vPC Peer-Links and core-facing interfaces on a single module, you should configure, using the command-line interface, a track object and a track list that is associated with the Layer 3 link to the core and on all vPC Peer-Links on both vPC peer devices. You use this configuration to avoid dropping traffic if that particular module goes down because when all the tracked objects on the track list go down, the system does the following:

- Stops the vPC primary peer device sending peer-keepalive messages, which forces the vPC secondary peer device to take over.
- Brings down all the downstream vPCs on that vPC peer device, which forces all the traffic to be rerouted in the access switch toward the other vPC peer device.

Once you configure this feature and if the module fails, the system automatically suspends all the vPC links on the primary vPC peer device and stops the peer-keepalive messages. This action forces the vPC secondary device to take over the primary role and all the vPC traffic to go to this new vPC primary device until the system stabilizes.

You should create a track list that contains all the links to the core and all the vPC Peer-Links as its object. Enable tracking for the specified vPC domain for this track list. Apply this same configuration to the other vPC peer device. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for information about configuring object tracking and track lists.

Note This example uses Boolean OR in the track list and forces all traffic to the vPC peer device only for a complete module failure. If you want to trigger a switchover when any core interface or vPC Peer-Link goes down, use a Boolean AND in the torack list below.

To configure a track list to switch over a vPC to the remote peer when all related interfaces on a single module fail, follow these steps:

1. Configure track objects on an interface (Layer 3 to core) and on a port channel (vPC Peer-Link).

```
switch(config-if)# track 35 interface ethernet 8/35 line-protocol
switch(config-track)# track 23 interface ethernet 8/33 line-protocol
switch(config)# track 55 interface port-channel 100 line-protocol
```

2. Create a track list that contains all the interfaces in the track list using the Boolean OR to trigger when all objects fail.

```
switch(config)# track 44 list boolean OR
switch(config-track)# object 23
switch(config-track)# object 35
switch(config-track)# object 55
switch(config-track)# end
```

3. Add this track object to the vPC domain:

```
switch(config)# vpc domain 1
switch(config-vpc-domain)# track 44
```

4. Display the track object:

```
switch# show vpc brief
Legend:
(*) - local vPC is down, forwarding via vPC peer-link
vPC domain id : 1
Peer status : peer adjacency formed ok
vPC keep-alive status : peer is alive
Configuration consistency status: success
vPC role : secondary
Number of vPCs configured : 52
Track object : 44
vPC Peer-link status
---------------------------------------------------------------------
id Port Status Active vlans
-- ---- ------ --------------------------------------------------
1 Po100 up 1-5,140
vPC status
----------------------------------------------------------------------------
id Port Status Consistency Reason Active vlans
-- ---- ------ ----------- -------------------------- ------------
```

```
    1 Po1 up success success 1-5,140
```

This example shows how to display information about the track objects:

```
switch# show track brief
Track Type Instance Parameter State Last
Change
23 Interface Ethernet8/33 Line Protocol UP 00:03:05
35 Interface Ethernet8/35 Line Protocol UP 00:03:15
44 List ----- Boolean
or UP 00:01:19
55 Interface port-channel100 Line Protocol UP 00:00:34
```

vPC Interactions with Other Features

vPC and LACP

LACP uses the system MAC address of the vPC domain to form the LACP Aggregation Group (LAG) ID for the vPC. (See the “Configuring Port Channels” chapter for information about LAG-ID and LACP.)

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

We recommend that you manually configure the system priority on the vPC Peer-Link devices to ensure that the vPC Peer-Link devices have a higher LACP priority than the downstream connected devices. A lower numerical value system priority means a higher LACP priority.

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

vPC Peer-Links and STP

Although vPCs provide a loop-free Layer 2 topology, STP is still required to provide a fail-safe mechanism to protect against any incorrect or defective cabling or possible misconfiguration. When you first bring up a vPC, 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. If the STP enhancements are already configured, they do not cause any problems for the vPC Peer-Links..

When you are running both MST and Rapid PVST+, ensure that the PVST simulation feature is correctly configured.

See the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide for information about STP enhancement features and PVST simulation.

Note You must configure a list of parameters to be identical on the vPC peer devices on both sides of the vPC Peer-Link. See the "Compatibility Parameters for vPC Interfaces" section for information about these required matched settings.

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

The primary vPC device synchronizes the STP state on the vPC secondary peer device using Cisco Fabric Services over Ethernet (CFSoE). See the "vPC and Orphan Ports" section for information about CFSoE.

The STP process for vPC also relies on the periodic keepalive messages to determine when one of the connected devices on the vPC Peer-Link fails. See the "Peer-Keepalive Link and Messages" section for information about these messages.

The vPC manager performs a proposal/handshake agreement between the vPC peer devices that set the primary and secondary devices and coordinates the two devices for STP. The primary vPC peer device then controls the STP protocol on both the primary and secondary devices. We recommend that you configure the primary vPC peer device as the STP primary root device and configure the secondary VPC device to be the STP secondary root device.

If the primary vPC peer device fails over to the secondary vPC peer device, there is no change in the STP topology.

The BPDUs uses the MAC address set for the vPC for the STP bridge ID in the designated bridge ID field. The vPC primary device sends these BPDUs on the vPC interfaces.

You must configure both ends of vPC Peer-Link with the identical STP configuration for the following parameters:

- STP global settings:
  - STP mode
  - STP region configuration for MST
  - Enable/disable state per VLAN
  - Bridge Assurance setting
  - Port type setting
  - Loop Guard settings
- STP interface settings:
  - Port type setting
  - Loop Guard
  - Root Guard

Note If any of these parameters are misconfigured, the Cisco NX-OS software suspends all interfaces in the vPC. Check the syslog and enter the **show vpc brief** command to see if the vPC interfaces are suspended.

Ensure that the following STP interface configurations are identical on both sides of the vPC Peer-Links or you may see unpredictable behavior in the traffic flow:

- BPDU Filter
- BPDU Guard
- Cost
- Link type
- Priority
- VLANs (PVRST+)

Note Display the configuration on both sides of the vPC Peer-Link to ensure that the settings are identical.

You can use the **show spanning-tree** command to display information about the vPC when that feature is enabled. See the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide for an example.

Note We recommend that you configure the ports on the downstream devices as STP edge ports. You should configure all host ports connected to a switch as STP edge ports. See the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide for more information about STP port types.

vPC Peer Switch

The vPC peer switch feature was added to Cisco NX-OS to address performance concerns around STP convergence. This feature allows a pair of Cisco Nexus 9000 Series devices to appear as a single STP root in the Layer 2 topology. This feature eliminates the need to pin the STP root to the vPC primary switch and improves vPC convergence if the vPC primary switch fails.

To avoid loops, the vPC Peer-Link is excluded from the STP computation. In vPC peer switch mode, STP BPDUs are sent from both vPC peer devices to avoid issues related to STP BPDU timeout on the downstream switches, which can cause traffic disruption.

This feature can be used with the pure peer switch topology in which the devices all belong to the vPC.

Note Peer-switch feature is supported on networks that use vPC and STP-based redundancy is not supported. If the vPC Peer-Link fail in a hybrid peer-switch configuration, you can lose traffic. In this scenario, the vPC peers use the same STP root ID as well as the same bridge ID. The access switch traffic is split in two with half going to the first vPC peer and the other half to the second vPC peer. With vPC Peer-Link failure, there is no impact to the north/south traffic but the east/west traffic is lost.

See the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide for information about STP enhancement features and Rapid PVST+.

vPC Peer-Gateway

You can configure vPC peer devices to act as the gateway even for packets that are destined to the vPC peer device's MAC address.

Use the **peer-gateway** command to configure this feature.

Note The **peer-gateway exclude-vlan** command that is used when configuring a VLAN interface for Layer 3 backup routing on vPC peer devices is not supported.

Some network-attached storage (NAS) devices or load balancers might have features that help to optimize the performances of particular applications. These features enable the device to avoid a routing-table lookup when responding to a request that originated from a host that is not locally attached to the same subnet. Such devices might reply to traffic using the MAC address of the sender Cisco Nexus 9000 Series device rather than the common HSRP gateway. This behavior is noncomplaint with some basic Ethernet RFC standards. Packets that reach a vPC device for the nonlocal router MAC address are sent across the vPC Peer-Link and could be dropped by the built in vPC loop avoidance mechanism if the final destination is behind another vPC.

The vPC peer-gateway capability allows a vPC switch to act as the active gateway for packets that are addressed to the router MAC address of the vPC peer. This feature enables local forwarding of packets without the need to cross the vPC Peer-Link. In this scenario, the feature optimizes use of the vPC Peer-Link and avoids potential traffic loss.

Configuring the peer-gateway feature must be done on both primary and secondary vPC peers and is nondisruptive to the operations of the device or to the vPC traffic. The vPC peer-gateway feature can be configured globally under the vPC domain submode.

When you enable this feature, Cisco NX-OS automatically disables IP redirects on all interface VLANs mapped over a vPC VLAN to avoid generation of IP redirect messages for packets switched through the peer gateway router.

Packets that arrive at the peer-gateway vPC device have their Time to Live (TTL) decremented, so that packets carrying a TTL of 1 might get dropped in transit due to TTL expiration. You should take this situation into account when the peer-gateway feature is enabled and particular network protocols that source packets with a TTL of 1 operate on a vPC VLAN.

vPC and ARP or ND

A feature was added to Cisco NX-OS to address table synchronization across vPC peers using the reliable transport mechanism of the Cisco Fabric Service over Ethernet (CFSoE) protocol. You must enable the **ip arp synchronize** and **ipv6 nd synchronize** commands to support faster convergence of address tables between the vPC peers. This convergence overcomes the delay that occurs in ARP table restoration for IPv4 or ND table restoration for IPv6 when the vPC Peer-Link port channel flaps or when a vPC peer comes back online.

vPC Multicast—PIM, IGMP, and IGMP Snooping

The Cisco NX-OS software for the Nexus 9000 Series devices supports the following on a vPC:

- PIM Any Source Multicast (ASM).

- PIM Source-Specific Multicast (SSM) .

Note The Cisco NX-OS software does not support Bidirectional (BIDR) on a vPC.

The software keeps the multicast forwarding state synchronized on both of the vPC peer devices. The IGMP snooping process on a vPC peer device shares the learned group information with the other vPC peer device through the vPC Peer-Link; the multicast states are always synchronized on both vPC peer devices. The PIM process in vPC mode ensures that only one of the vPC peer devices forwards the multicast traffic to the receivers.

Each vPC peer is a Layer 2 or Layer 3 device. Multicast traffic flows from only one of the vPC peer devices. You might see duplicate packets in the following scenarios:

- Orphan hosts
- When the source and receivers are in the Layer 2 vPC cloud in different VLANs with multicast routing enabled and a vPC member link goes down.

You might see negligible traffic loss in the following scenarios:

- When you reload the vPC peer device that is forwarding the traffic.
- When you restart PIM on the vPC peer device that is forwarding the traffic.

Overall multicast convergence times are scale and vPC role change / PIM restart duration dependent.

Ensure that you dual-attach all Layer 3 devices to both vPC peer devices. If one vPC peer device goes down, the other vPC peer device continues to forward all multicast traffic normally.

The following outlines vPC PIM and vPC IGMP/IGMP snooping:

- vPC PIM—The PIM process in vPC mode ensures that only one vPC peer device forwards multicast traffic. The PIM process in vPC mode synchronizes the source state with both vPC peer devices and elects which vPC peer device forwards the traffic.
- vPC IGMP/IGMP snooping—The IGMP process in vPC mode synchronizes the designated router (DR) information on both vPC peer devices. Dual DRs are available for IGMP when you are in vPC mode. Dual DRs are not available when you are not in vPC mode, because both vPC peer devices maintain the multicast group information between the peers.

Note A PIM adjacency between a Switched Virtual Interface (SVI) on a vPC VLAN (a VLAN that is carried on a vPC Peer-Link) and a downstream device is not supported; this configuration can result in dropped multicast packets. If a PIM neighbor relationship is required with a downstream device, a physical Layer 3 interface must be used on the Nexus switches instead of a vPC SVI.

For SVIs on vPC VLANs, only one PIM adjacency is supported, which is with the vPC peer switch. PIM adjacencies over the vPC Peer-Link with devices other than the vPC peer switch for the vPC-SVI are not supported.

You should enable or disable IGMP snooping identically on both vPC peer devices, and all the feature configurations should be identical. IGMP snooping is on by default.

Note The following commands are not supported in vPC mode:

- **ip pim spt-threshold infinity**
- **ip pim use-shared-tree-only**

See the *Cisco Nexus 9000 Series NX-OS Multicast Routing Configuration Guide* for more information about multicasting.

Multicast PIM Dual DR (Proxy DR)

By default, a multicast router sends PIM joins upstream only if it has interested receivers. These interested receivers can either be IGMP hosts (they communicate through IGMP reports) or other multicast routers (they communicate through PIM joins).

In the Cisco NX-OS vPC implementation, PIM works in dual designated router (DR) mode. That is, if a vPC device is a DR on a vPC SVI outgoing interface (OIF), its peer automatically assumes the proxy DR role. IGMP adds an OIF (the report is learned on that OIF) to the forwarding if the OIF is a DR. With dual DRs, both vPC devices have an identical (*,G) entry with respect to the vPC SVI OIFs as shown in this example:

```
VPC Device1:
------------
(*,G)
oif1 (igmp)
VPC Device2:
------------
(*,G)
oif1 (igmp)
```

IP PIM PRE-BUILD SPT

When the multicast source is in a Layer 3 cloud (outside the vPC domain), one vPC peer is elected as the forwarder for the source. This forwarder election is based on the metrics to reach the source. If there is a tie, the vPC primary is chosen as the forwarder. Only the forwarder has the vPC OIFs in its associated (S,G) and the nonforwarder (S,G) has 0 OIFs. Therefore, only the forwarder sends PIM (S,G) joins toward the source as shown in this example:

```
VPC Device1 (say this is Forwarder for Source 'S'):
------------
(*,G)
oif1 (igmp)
(S,G)
oif1 (mrib)
VPC Device2:
------------
(*,G)
oif1 (igmp)
(S,G)
NULL
```

In the case of a failure (for example, a Layer 3 Reverse Path Forwarding (RPF) link on the forwarder becomes inoperable or the forwarder gets reloaded), if the current nonforwarder ends up becoming the forwarder, it

has to start sending PIM joins for (S,G) toward the source to pull the traffic. Depending upon the number of hops to reach the source, this operation might take some time (PIM is a hop-by-hop protocol).

To eliminate this issue and get better convergence, use the **ip pim pre-build-spt** command. This command enables PIM send joins even if the multicast route has 0 OIFs. In a vPC device, the nonforwarder sends PIM (S,G) joins upstream toward the source. The downside is that the link bandwidth upstream from the nonforwarder gets used for the traffic that is ultimately dropped by it. The benefits that result with better convergence far outweigh the link bandwidth usage. Therefore, we recommend that you use this command if you use vPCs.

vPC Peer-Links and Routing

The First Hop Redundancy Protocols (FHRPs) interoperate with vPCs. The Hot Standby Routing Protocol (HSRP), and Virtual Router Redundancy Protocol (VRRP) all interoperate with vPCs. We recommend that you dual-attach all Layer 3 devices to both vPC peer devices.

The primary FHRP device responds to ARP requests, even though the secondary vPC device forwards the data traffic.

To simplify initial configuration verification and vPC/HSRP troubleshooting, you can configure the primary vPC peer device with the FHRP active router highest priority.

In addition, you can use the priority command in the if-hsrp configuration mode to configure failover thresholds for when a group state enabled on a vPC Peer-Link is in standby or in listen state. You can configure lower and upper thresholds to prevent the interface from going up and down.

VRRP acts similarly to HSRP when running on vPC peer devices. You should configure VRRP the same way that you configure HSRP.

When the primary vPC peer device fails over to the secondary vPC peer device, the FHRP traffic continues to flow seamlessly.

We recommend that you configure routing adjacency between the two vPC peer devices to act as a backup routing path. If one vPC peer device loses Layer 3 uplinks, the vPC can redirect the routed traffic to the other vPC peer device and leverage its active Layer 3 uplinks.

You can configure the inter-switch link for a backup routing path in the following ways:

- Create a Layer 3 link between the two vPC peer devices.
- Use the non-VPC VLAN trunk with a dedicated VLAN interface.
- Use a vPC Peer-Link with a dedicated VLAN interface.

We do not recommend that you configure the burnt-in MAC address option (use-bia) for HSRP or manually configure virtual MAC addresses for any FHRP protocol in a vPC environment because these configurations can adversely affect vPC load balancing. The HSRP use-bia option is not supported on vPCs. When you are configuring custom MAC addresses, you must configure the same MAC address on both vPC peer devices.

You can use the **delay restore** command to configure a restore timer that delays the vPC coming back up until after the peer adjacency forms and the VLAN interfaces are back up. This feature enables you to avoid packet drops when the routing tables might not be converged before the vPC is once again passing traffic. Use the **delay restore** command to configure this feature.

To delay the VLAN interfaces on the restored vPC peer device from coming up, use the **interfaces-vlan** option of the **delay restore** command.

See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for more information about FHRPs and routing.

Configuring Layer 3 Backup Routes on a vPC Peer-Link

You can use VLAN network interfaces on the vPC peer devices to link to Layer 3 of the network for such applications as HSRP and PIM. Ensure that you have a VLAN network interface configured on each peer device and that the interface is connected to the same VLAN on each device. Also, each VLAN interface must be in the same administrative and operational mode. For more information about configuring VLAN network interfaces, see the "Configuring Layer 3 Interfaces" chapter.

If a failover occurs on the vPC Peer-Link, the VLAN interfaces on the vPC peer devices are also affected. If a vPC Peer-Link fails, the system brings down associated VLAN interfaces on the secondary vPC peer device.

You can ensure that specified VLAN interfaces do not go down on the vPC secondary device when the vPC Peer-Link fails.

CFSoE

The Cisco Fabric Services over Ethernet (CFSoE) is a reliable state transport mechanism that is used 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.

The CFSoE transport is local to each VDC.

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. You must enable CFSoE for vPC functionality. If you do enter either of these commands with vPC 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.

CFS also transports data over TCP/IP. See the Cisco Nexus 9000 Series NX-OS System Management Configuration Guide for more information about CFS over IP.

Note The software does not support CFS regions.

vPC and Orphan Ports

When a device that is not vPC-capable connects to each peer, the connected ports are known as orphan ports because they are not members of a vPC. The device's link to one peer will be active (forwarding) and the other link will be standby (blocking) due to STP.

If a vPC Peer-Link failure or restoration occurs, an orphan port's connectivity might be bound to the vPC failure or restoration process. For example, if a device's active orphan port connects to the secondary vPC

peer, the device loses any connections through the primary peer if a vPC Peer-Link failure occurs and the vPC ports are suspended by the secondary peer. If the secondary peer were to also suspend the active orphan port, the device's standby port becomes active, provides a connection to the primary peer, and restores connectivity. You can configure in the CLI that specific orphan ports are suspended by the secondary peer when it suspends its vPC ports and are restored when the vPC is restored.

Virtualization Support

All ports in a given vPC must be in the same VDC. This version of the software supports only one vPC domain per VDC. You can use the numbers from 1 to 4096 in each VDC to number the vPC.

vPC Recovery After an Outage

In a data center outage, both the vPC peer in vPC domain get reloaded. Occasionally only one peer can be restored. With no functioning peer-keepalive or vPC Peer-Link, the vPC cannot function normally, a method might be available to allow vPC services to use only the local ports of the functional peer.

Autorecovery

You can configure the Cisco Nexus 9000 Series device to restore vPC services when its peer fails to come online by using the **auto-recovery** command. You must save this setting in the startup configuration. On reload, if the vPC Peer-Link is down and three consecutive peer-keepalive messages are lost, the secondary device assumes the primary STP role and the primary LACP role. The software reinitializes the vPCs, bringing up its local ports. Because there are no peers, the consistency check is bypassed for the local vPC ports. The device elects itself to be the STP primary regardless of its role priority and also acts as the primary device for LACP port roles.

Autorecovery reload-delay

vPC peer auto recovery can be delayed using **auto-recovery reload-delay** command. Auto-recovery reload-delay time is used on peer that comes up first. The **reload-delay time** command is used to wait for both peers to recover and to keep existing roles before auto recovery starts. The device then resumes primary role to recovered switch.

vPC Peer Roles After a Recovery

When the other peer device completes its reload and adjacency forms, the following process occurs:

1. The first vPC peer maintains its current role to avoid any transition reset to other protocols. The peer accepts the other available role.
2. When an adjacency forms, consistency checks are performed and appropriate actions are taken.

High Availability

During an In-Service Software Upgrade (ISSU), the software reload process on the first vPC device locks its vPC peer device by using CFS messaging over the vPC communications channel. Only one device at a time is upgraded. When the first device completes its upgrade, it unlocks its peer device. The second device then performs the upgrade process, locking the first device as it does so. During the upgrade, the two vPC devices

temporarily run different releases of Cisco NX-OS, however the system functions correctly because of its backward compatibility support.

Note See the Cisco Nexus 9000 Series NX-OS High Availability and Redundancy Guide for complete information about high-availability features.

vPC Forklift Upgrade Scenario

The following procedure describes a scenario of migrating pair of Cisco Nexus 9500 switches in a vPC domain to a different pair of Cisco Nexus 9500 switches with a same type of line cards. Migrating from Cisco Nexus 9504 switches to Cisco Nexus 9508 switches for the need of more interfaces is a typical example of such migration. The following migration scenarios are not supported:

- Migration of Cisco Nexus 9500 switches with a different set of line cards. For example, from a Cisco Nexus 9500 switches with N9K-X94xx line card to Cisco Nexus 9500 switches with N9K-X97xx line card.
- Migration between different generations of Cisco Nexus 9300 switches. For example, migration from Cisco Nexus N9K-C9372PX to Cisco Nexus N9K-93180YC-EX switches
- Having different generations of Cisco Nexus 9000 switches in a vPC domain is not supported

Considerations for a vPC forklift upgrade:

- vPC Role Election and Sticky-bit

  By default, the Cisco NX-OS software uses the lowest MAC address to elect the primary device. However, if the role priority is set, then the device with the lowest priority will be elected as the primary device. When the primary device is reloaded, the system comes back online and connectivity to the vPC secondary device (now the operational primary) is restored. The operational role of the secondary device (operational primary) does not change (to avoid unnecessary disruptions). This behavior is achieved with a sticky-bit, where the sticky information is not saved in the startup configuration. This method makes the device that is up and running win over the reloaded device. Hence, the vPC primary becomes the vPC operational secondary. Sticky-bit is also set when a vPC node comes up with vPC Peer-Link and peer-keepalive down and it becomes primary after the auto recovery period.

- vPC Delay Restore

  The delay restore timer is used to delay the vPC from coming up on the restored vPC peer device after a reload when the peer adjacency is already established.

  To delay the VLAN interfaces on the restored vPC peer device from coming up, use the **interfaces-vlan** option of the **delay restore** command.

- vPC Auto-Recovery

  During a data center power outage when both vPC peer switches go down, if only one switch is restored, the auto-recovery feature allows that switch to assume the role of the primary switch and the vPC links come up after the auto-recovery time period. The default auto-recovery period is 240 seconds.

The following example is a migration scenario that replaces vPC peer nodes Node1 and Node2 with New_Node1 and New_Node2.

	Migration Step	Expected Behavior	Node1 Configured role (Ex: role priority 100)	Node1 Operational role	Node2 Configured role (Ex: role priority 200)	Node2 Operational role
1	Initial state	Traffic is forwarded by both vPC peers – Node1 and Node2. Node1 is primary and Node2 is secondary.	primary	Primary Sticky bit: False	secondary	Secondary Sticky bit: False
2	Node2 replacement – Shut all vPCs and uplinks on Node2. vPC Peer-Link and vPC peer-keepalive are in administrative up state.	Traffic converged on Primary vPC peer Node1.	primary	Primary Sticky bit: False	secondary	Secondary Sticky bit: False
3	Remove Node2.	Node1 will continue to forward traffic.	primary	Primary Sticky bit: False	n/a	n/a
4	Configure New_Node2. Copy the configuration to startup config. vPC vPC Peer-Link and peer-keepalive in administrative up state. Power off New_Node2. Make all connections. Power on New_Node2.	New_Node2 will come up as secondary. Node1 continue to be primary. Traffic will continue to be forwarded on Node01.	primary	Primary Sticky bit: False	secondary	Secondary Sticky bit: False
5	Bring up all vPCs and uplink ports on New_Node2.	Traffic will be forwarded by both Node 1 and New_Node2.	primary	Primary Sticky bit: False	secondary	Secondary Sticky bit: False
6	Node1 replacement - Shut vPCs and uplinks on Node1.	Traffic will converge on New_Node2.	primary	Primary Sticky bit: False	secondary	Secondary Sticky bit: False

	Migration Step	Expected Behavior	Node1 Configured role (Ex: role priority 100)	Node1 Operational role	Node2 Configured role (Ex: role priority 200)	Node2 Operational role
7	Remove Node1.	New_Node2 will become secondary, operational primary and sticky bit will be set to True.	n/a	n/a	secondary	Primary Sticky bit: True
8	Configure New_Node1. Copy running to startup. Power off the new Node1. Make all connections. Power on New_Node1.	New_Node1 will come up as primary, operational secondary.	primary	Secondary Sticky bit: False	secondary	Primary Sticky bit: True
9	Bring up all vPCs and uplink ports on New_Node1.	Traffic will be forwarded by both New Node1 and new Node2.	primary	Secondary Sticky bit: False	secondary	Primary Sticky bit: True

Note If you prefer to have the configured secondary node as the operational secondary and the configured primary as the operational primary, then Node2 can be reloaded at the end of the migration. This is optional and does not have any functional impact.

Guidelines and limitations

These are configuration guidelines and limitations for vPC.

- All ports for a given vPC must be in the same VDC.
- You must enable vPCs before you can configure them.
- Only Layer 2 port channels can be in vPCs.
- You must configure both vPC peer devices; the configuration is not sent from one device to the other.
- You may experience minimal traffic disruption while configuring vPCs on existing port-channels.
- The software does not support CFS regions.
- The STP port cost is fixed to 200 in a vPC environment.
- To configure multilayer (back-to-back) vPCs, you must assign unique vPC domain ID for each respective vPC.

There might be duplicate multicast streams with Layer 3 links and with the back-to-back vPC when:

- SVI is configured on all four switches that are part of a back-to-back vPC.
- There are additional L3 links connecting the four switches which are part of vPC.
- PIM is enabled on all SVIs and on the L3 links between switches.

To prevent the duplicate streams, remove SVIs or the PIM configuration from one of the vPC switch pairs.

- The software does not support BIDR PIM, SSM on vPCs.
- The software does not support DHCP snooping, DAI, or IPSG in a vPC environment; DHCP Relay is supported.
- Peer-switch can only be configured if both VPC peers share the same priority, and are root for all VLANs or MST instances. Peer-switch cannot be configured if at least one VLAN or MST instance is not root.
- FEX-AA (dual-homed FEX) and FEX-ST (FEX straight-thru) topologies (FEX-AA and FEX-ST) are supported. The following parent switch combinations are not supported:
  - Cisco Nexus 9300-EX and 9300 switches.
  - Cisco Nexus 9300 and 9500 switches.
  - Cisco Nexus 9300-EX and 9500 switches.
- Starting with Cisco NX-OS Release 9.3(5) Cloud Scale based TOR switches can forward TTL=1 packet destined to vPC peer in hardware/data plane. It is recommended to use one of these releases or later releases for a seamless operation of the feature.
- When you configure a vPC pair for STP priority, you must set the same priority level for both the vPC peer switches in order to get both vPC peers to work as STP root.
- **show** commands with the **internal** keyword are *not* supported.
- Cisco Nexus 9000 Series switches do *not* support NAT on vPC topology.
- Starting from Cisco NX-OS Release 9.2(1,) the **show vpc consistency-checker** command is *not* available on Cisco Nexus 9000 switches.
- Starting from Cisco NX-OS Release 9.2(1,) the **delay restore interface-bridge-domain** and **peer-gateway exclude-bridge-domain** commands are *not* available on Cisco Nexus 9500-R platform switches.
- We recommend that you configure all the port channels in the vPC using LACP with the interfaces in active mode.
- The **vpc orphan-ports suspend** command also applies to ports in non-vPC VLANs and Layer 3 ports. However, it is recommended to be used with ports in VPC VLANs.
- To form a supported vPC domain, ensure that the following is taken care:
  - For Cisco Nexus 9300 Series switches, both switches must be of the exact same model.
  - For Cisco Nexus 9500 Series switches, both switches must consist of the same models of line cards, fabric modules, supervisor modules, and system controllers inserted in the same slots of the chassis.
- All the devices that are attached to a vPC domain through a vPC must be dual homed.

- You must run the commands **lacp suspend-individual** and **lacp mode delay** to PXE boot the servers that are connected Cisco Nexus 9000 switches via vPC.

Guidelines for vPC Peer Link

- You must configure the peer-keepalive link and adjacency between peers must be formed before the system can establish the vPC Peer-Link.
- You must ensure that all the necessary configuration parameters are compatible on both sides of the vPC Peer-Link. See the *Compatibility Parameters for vPC Interfaces* section for information about compatibility recommendations.
- vPC Peer-Link by default has set MTU of 9216.
- To accommodate increased traffic when the vPC goes down and traffic needs to cross the vPC Peer-Link, it is a best practice to use multiple high bandwidth interfaces (such as the 40G interfaces for the Cisco Nexus 9000) across linecards for the vPC Peer-Link.
- If you configure open shortest path first (OSPF) in a vPC environment, use the following timer commands in router configuration mode on the core switch to ensure fast OSPF convergence when a vPC Peer-Link is shut down:

```
switch (config-router)# timers throttle spf 1 50 50
switch (config-router)# timers lsa-arrival 10
```

  See the *Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide* for further details about OSPF.
- Jumbo frames are enabled by default on the vPC Peer-Link.
- LACP configuration on the vPC port-channel must be consistent on both the Cisco Nexus switches across a vPC Peer-Link.
- When **peer-switch** features are configured under **vpc domain** configuration mode on two Cisco Nexus 9000 Series switches, the spanning-tree root changes even for VLANs that are not enabled on the vPC Peer-Link. Both the switches act as one system with one MAC address as the bridge address. This is true even for non-vPC mst-instance or VLANs. Therefore, a non vPC Peer-Link between the two switches gets blocked as a backup link. This is an expected behavior.
- The first generation Broadcom based Nexus 9300 series switches and Nexus 9500 series linecards does not support Policy Based Routing (PBR) route map with **set ip next-hop** configuration for egress interfaces as the vPC Peer-Link for TCAM regions allocated for vPC convergence.

  This limitation does not apply to cloud scale based Nexus 9000 series devices such as Cisco Nexus 9200 switches, 9300 switches with EX/FX/FX2 line-cards and Nexus 9500 platform switches with 9700-EX/FX line-cards.

Guidelines for vPC STP Hitless Role

- vPC role change can be performed from either of the peer devices.
- If the original secondary device has higher role priority value than the original primary device, role swapping cannot be performed. Change the role priority on either vPC device so that the value of the original secondary device is lower than the original primary one. To view the existing role of a device, use the show vpc role command on local and peer switch.

- Always check the existing configured role priority before configuring vPC hitless role changefeature In a vPC domain, enable the peer-switch command, where both vPC peers have same STP priorities, and ensure it is operational before issuing a role change. If you do not enable the peer-switch command, it can lead to convergence issues. Use **show spanning-tree summary | grep peer** command to verify whether the peer vPC switch is operational or not.

Guidelines for vPC peers in HSRP

- When migrating from a pair of spine nodes to a pair of Cisco Nexus 9000 devices, the HSRP priority should be configured so that the Cisco Nexus 9000 vPC peers are in Active/Standby state. There is no support for Cisco Nexus 9000 vPC peers in HSRP state to be in Active/Listen state, or Standby/Listen state
- When using vPCs, we recommend that you use default timers for FHRP (HSRP, VRRP), and PIM configurations. Using aggressive timers in vPC configurations has no advantage in convergence times.
- BFD for VRRP/HSRP is not supported in a vPC environment.
- Having the same Hot Standby Router Protocol (HSRP)/Virtual Router Redundancy Protocol (VRRP) group on all nodes on a double sided vPC is supported.
- When migrating from a pair of spine nodes to a pair of Cisco Nexus 9000 devices, the HSRP priority should be configured so that the Cisco Nexus 9000 vPC peers are in Active/Standby state. There is no support for Cisco Nexus 9000 vPC peers in HSRP state to be in Active/Listen state, or Standby/Listen state.

Guidelines for Layer 3 over vPC

- Layer 3 over vPC is supported on Cisco Nexus 9000 Series switches for Layer 3 unicast communication only.

  Layer 3 over vPC is not supported for Layer 3 multicast traffic. For more information please refer to the *Best Practices for Layer 3 and vPC Configuration* section
- By default Layer 3 vPC forwards all the packets (with TTL=1) destined for the peer vPC node. OSPF/BGP can flap due to this forwarding. You need to carve the ing-sup TCAM to size 768 in order to make the switch hardware forward. Make sure to reload the switch after the TCAM carving. An example is listed below.

  ```
  show hardware access-list tcam region | gr ing-sup
       Ingress SUP [ing-sup] size = 768
  ```

  Cisco NX-OS Release 9.3(4) has this default behavior though a TCAM re-carving option is available for the hardware redirect of the packets to vPC peer for Cloud Scale based TOR switches. This requires allocating at least 768 space for ing-sup region and requires reload and has operational overhead.
- The default behavior with Layer 3 peer-router and TTL=1 packet destined to IP of vPC peer is to punt packet to CPU and then forward the software to vPC peer. This is applicable to the Cloud Scale based EOR switches.
- You may see the following behavior with unicast packets when you configure Layer 3 peer-router with Cloud Scale ASIC based switches:
  - Unicast packets with TTL=0 destined to vPC peer node, will be forwarded to the peer.
  - Unicast packets with TTL=0 are not dropped by the peer, it gets punted to SUP instead.

- Unicast packets with TTL=1 and TTL=0 destined to VPC peer node can be software forwarded and hardware forwarded. So duplicate packets are seen in the peer node.

- Beginning with Cisco NX-OS Release 9.3(9), a syslog is created when peer-gateway and layer 3 peer-router commands are not configured on both the vPC peers in the vPC domain.

Guidelines for vPC During Upgrade

- vPC peers must run the same Cisco NX-OS release. During a software upgrade, you must upgrade the primary vPC peer first.
- Before performing a non-disruptive upgrade, you must make sure that both vPC peers are in the same mode (regular ISSU mode or enhance ISSU mode).

Note vPC peering between an enhanced ISSU mode (boot mode lxc) configured switch and a non-enhanced ISSU mode switch is *not* supported.

Best Practices for Layer 3 and vPC Configuration

This section describes best practices for using and configuring Layer 3 with vPC.

Layer 3 and vPC Configuration Overview

When a Layer 3 device is connected to a vPC domain through a vPC, it has the following views:

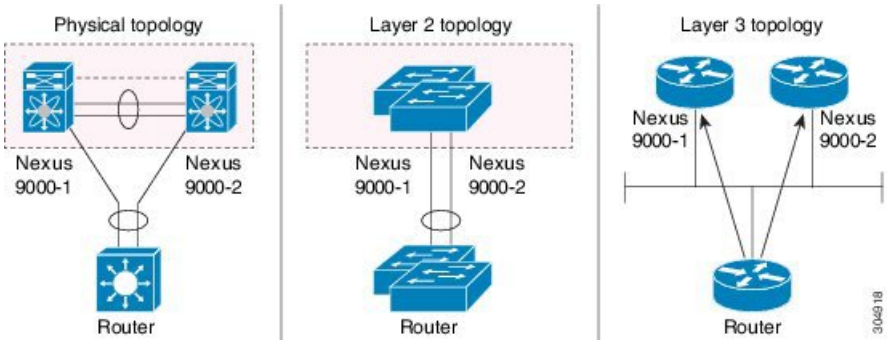
- At Layer 2, the Layer 3 device sees a unique Layer 2 switch presented by the vPC peer devices.
- At Layer 3, the Layer 3 device sees two distinct Layer 3 devices (one for each vPC peer device).

vPC is a Layer 2 virtualization technology, so at Layer 2, both vPC peer devices present themselves as a unique logical device to the rest of the network.

There is no virtualization technology at Layer 3, so each vPC peer device is seen as a distinct Layer 3 device by the rest of the network.

The following figure illustrates the two different Layer 2 and Layer 3 views with vPC.

Figure 17: Different Views for vPC Peer Devices

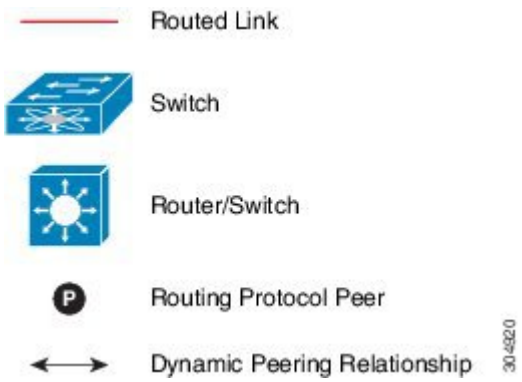


Supported Topologies for Layer 3 and vPC

This section contains examples of Layer 3 and vPC network topologies.

There are two approaches for Layer 3 and vPC interactions. The first one is by using dedicated Layer 3 links to connect the Layer 3 devices to each vPC peer device. The second one is by allowing the Layer 3 devices to peer with the SVIs defined on each of the vPC peer device, on a dedicated VLAN that is carried on the vPC connection. The following sections describe all the supported topologies leveraging the elements that are described in the legends in the following figure.

Figure 18: Legend

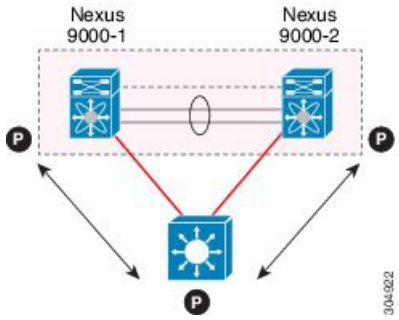


Peering with an External Router Using Layer 3 Links

This example shows a topology that uses Layer 3 links to connect a Layer 3 device to the Cisco Nexus 9000 switches that are part of the a vPC domain

Note Interconnecting the two entities together in this way allows to support Layer 3 unicast and multicast communication.

Figure 19: Peering with an External Router Using Layer 3 Links



Layer 3 devices can initiate Layer 3 routing protocol adjacencies with both vPC peer devices.

One or multiple Layer 3 links can be used to connect a Layer 3 device to each vPC peer device. Cisco Nexus 9000 series devices support Layer 3 Equal Cost Multipathing (ECMP) with up to 16 hardware load-sharing paths per prefix. Traffic from a vPC peer device to a Layer 3 device can be load-balanced across all the Layer 3 links interconnecting the two devices together.

Using Layer 3 ECMP on the Layer 3 device can effectively use all Layer 3 links from the device to the vPC domain. Traffic from a Layer 3 device to the vPC domain can be load-balanced across all the Layer 3 links interconnecting the two entities together.

Follow these guidelines when connecting a Layer 3 device to the vPC domain using Layer 3 links:

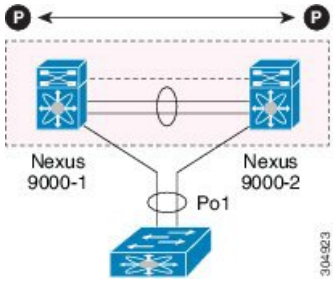
- Use separate Layer 3 links to connect Layer 3 devices to the vPCdomain. Each link represents a point-to-point Layer 3 connection and should get assigned an IP address taken from a small IP subnet (/30 or /31).
- If the Layer 3 peering is required for multiple VRFs, it is recommended to define multiple sub-interfaces, each mapped to an individual VRF.

Peering Between vPC Devices for a Backup Routing Path

This example shows peering between the two vPC peer devices with a Layer 3 backup routed path. If the Layer 3 uplinks on vPC peer device 1 or vPC peer device 2 fail, the path between the two peer devices is used to redirect traffic to the switch that has the Layer 3 uplinks in the up state.

The Layer 3 backup routing path can be implemented using a dedicated interface VLAN (such as SVI) over the vPC Peer-Link or by using dedicated Layer 2 or Layer 3 links across the two vPC peer devices.

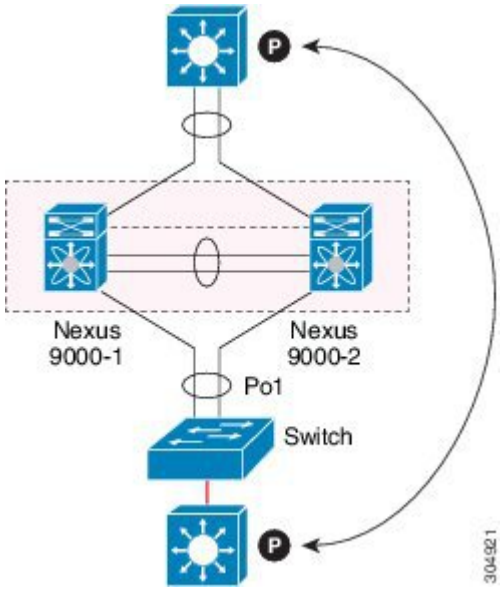
Figure 20: Peering Between vPC Devices for a Backup Routing Path



Direct Layer 3 Peering Between Routers

In this scenario, the Nexus 9000 devices part of the vPC domain are simply used as a Layer 2 transit path to allow the routers connected to them to establish Layer 3 peering and communication.

Figure 21: Peering Between Routers



The Layer 3 devices can peer with each other in following two methods. Peering also depends on the specific device deployed for this role.

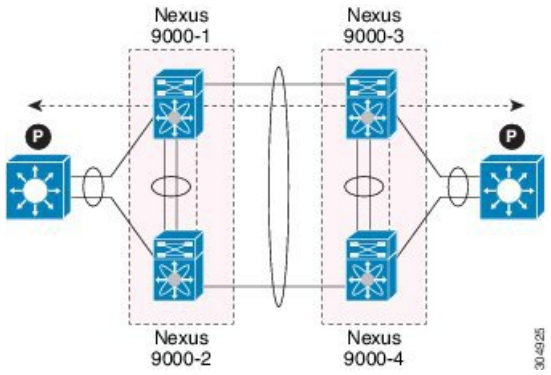
- Defining a VLAN network interface (SVI) for a VLAN that is extended between the Layer 3 devices through the intermediate Cisco Nexus 9000 vPC peer switches.
- Defining a Layer 3 port-channel interface on each Layer 3 device and establishing a point-to-point Layer 3 peering.

Note In deployments where the Layer 3 peering must be established for multiple VRFs, the first method require the definition on the Layer 3 devices of a VLAN (and SVI) per VRF. For the second method, it is possible to create a Layer 3 port-channel subinterface per VRF

Peering Between Two Routers with vPC Devices as Transit Switches

This example is similar to the peering between routers topology. In this case also, the Cisco Nexus 9000 devices that are part of the same vPC domain are only used as Layer 2 transit paths. The difference here is that there are two pairs of Cisco Nexus 9000 switches. Each switch that is connected with a Layer 3 device using a vPC connection, also establishes a back-to-back vPC connection between them. The difference is that the vPC domains are only used as Layer 2 transit paths.

Figure 22: Peering Between Two Routers with vPC Devices as Transit Switches



This topology is commonly used when you want to establish connectivity between separate data centers that are interconnected with direct links (dark fibers or DWDM circuits). The two pairs of Cisco Nexus 9000 switches, in this case, provide only Layer 2 extension services, allowing the Layer 3 devices to peer with each other at Layer 3.

Peering with an External Router on Parallel Interconnected Routed Ports

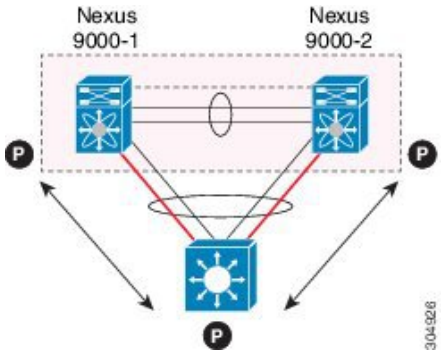
When you require both routed and bridged traffic, use individual Layer 3 links for routed traffic and a separate Layer 2 port-channel for bridged traffic, as shown in following figure.

The Layer 2 links are used for bridged traffic (traffic staying in the same VLAN) or inter-VLAN traffic (assuming vPC domain hosts the interface VLAN and associated HSRP configuration).

The Layer 3 links are used for routing protocol peering adjacency with each vPC peer device.

The purpose of this topology is to attract specific traffic to go through the Layer 3 device. Layer 3 links are also used to carry routed traffic from a Layer 3 device to the vPC domain.

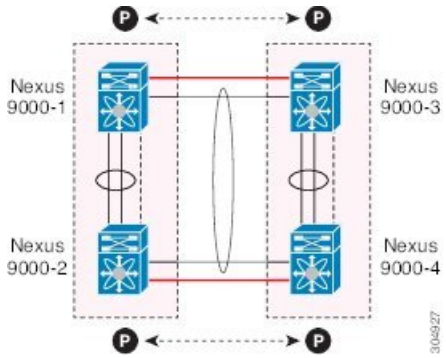
Figure 23: Peering with an External Router on Parallel Interconnected Routed Ports



Peering between vPC Switch Pairs on Parallel Interconnected Routed Ports

An alternative design to what is shown in the previous section (Peering Between Two Routers with vPC Devices as Transit Switches), uses two pairs of Cisco Nexus 9000 switches that are deployed in each data center for providing both Layer 2 and Layer 3 extension services. When routing protocol peering adjacency is required to be established between the two pairs of Cisco Nexus 9000 devices, the best practice is to add dedicated Layer 3 links between the two sites as shown in the following example.

Figure 24: Peering Over a vPC Interconnection on Parallel Interconnected Routed Ports



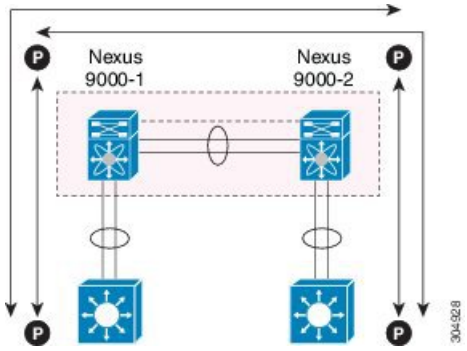
The back-to-back vPC connection between the two data centers carry bridged traffic or inter-VLAN traffic while the dedicated Layer 3 links carry the routed traffic across the two sites.

Peering Over a PC Interconnection and Dedicated Interswitch Link Using non-vPC VLAN

This example shows when the Layer 3 device is single-attached to the vPC domain, you can use a non-vPC VLAN with a dedicated inter-switch link to establish the routing protocol peering adjacency between the Layer 3 device and each vPC peer device. However, the non-vPC VLAN must be configured to use a static MAC that is different than the vPC VLAN.

Note Configuring the vPC VLAN (and vPC Peer-Link) for this purpose is not supported.

Figure 25: Peering Over a PC Interconnection and Dedicated Interswitch Link Using non-vPC VLAN

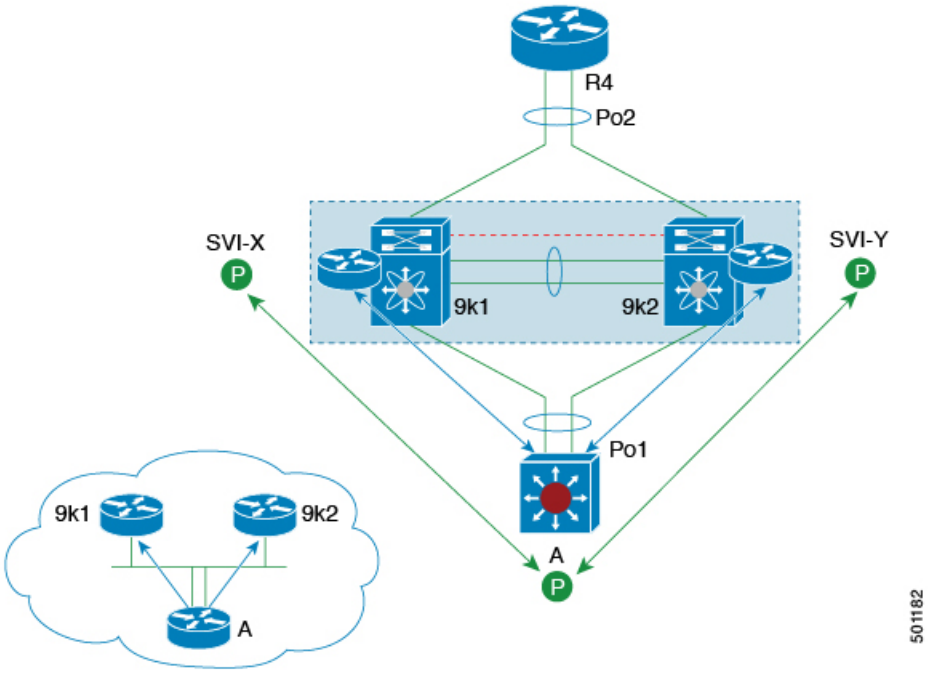


Peering Directly Over a vPC Connection

Beginning with Cisco NX-OS Release 7.0(3)I5(1), an alternative method has been introduced to establish Layer 3 peering between a Layer 3 router and a pair of Cisco Nexus 9000 vPC switches.

Note Peering directly over a vPC connection is supported only for Layer 3 unicast communication but not for Layer 3 multicast traffic. If you require Layer 3 multicast, you must establish peering over dedicated Layer 3 links

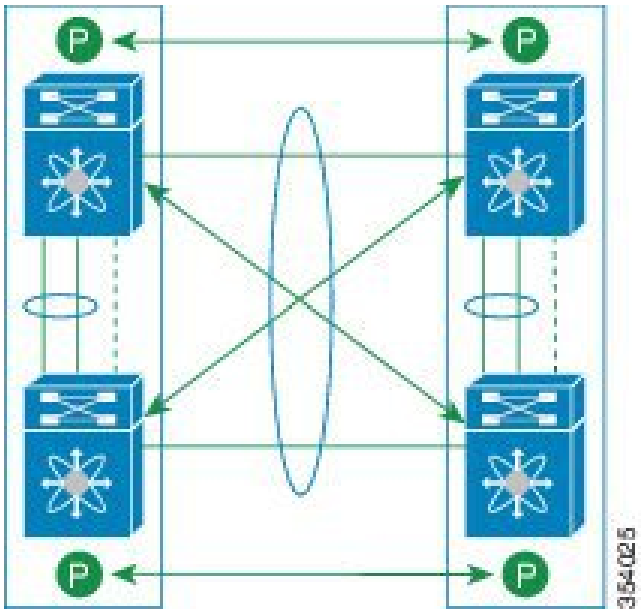
Figure 26: Supported: Peering Over a vPC Interconnection Where the Router Peers with Both the vPC Peers.



In this scenario, the Layer 3 peering between the external router and the Cisco Nexus 9000 switches that are part of a same vPC domain is established directly on a VLAN carried on the vPC connection. The external router in this case peers with SVI interfaces defined on each vPC device. As for the scenario shown in previous figure 12, the external router could use an SVI or a Layer 3 Port-Channel to peer with the vPC devices (multiple SVIs or Port-Channel subinterfaces could be used for a multi-VRF deployment).

This deployment model requires configuring **layer3 peer-router** command as part of the vPC domain. You can adopt the same approach for establishing Layer 2 and Layer 3 connectivity on a vPC back-to-back connection established between two separate pairs of vPC switches.

Figure 27: Supported: Peering Over a vPC Interconnection Where Each Nexus Device Peers with Two vPC Peers.



In this deployment model, SVI interfaces in the same VLAN is configured on all the four Cisco Nexus 9000 switches to establish routing peering and connectivity between them.

Default Settings

The following table lists the default settings for vPC parameters.

Table 18: 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

Note You must use these procedures on both devices on both sides of the vPC Peer-Link. You configure both of the vPC peer devices using these procedures.

This section describes how to configure vPCs using the command-line interface (CLI).

Note If you are familiar with the Cisco IOS CLI, be aware that the Cisco NX-OS commands for this feature might differ from the Cisco IOS commands that you would use.

Enabling vPCs

You must enable the feature vPC before you can configure and use vPCs.

SUMMARY STEPS

1. **configure terminal**
2. **feature vpc**
3. **exit**
4. **show feature**
5. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**feature vpc** **Example:** `switch(config)# feature vpc`	Enables vPCs on the device.

Step 3	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits global configuration mode.

Step 4	**show feature** **Example:** `switch# show feature`	(Optional) Displays which features are enabled on the device.
Step 5	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enable the vPC feature:

```
switch# configure terminal
switch(config)# feature vpc
switch(config)# exit
switch(config)#
```

Disabling vPCs

Note When you disable the vPC functionality, the device clears all the vPC configurations.

SUMMARY STEPS

1. **configure terminal**
2. **no feature vpc**
3. **exit**
4. **show feature**
5. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**no feature vpc** **Example:** `switch(config)# no feature vpc`	Disables vPCs on the device.

Step 3	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits global configuration mode.

Step 4	**show feature** **Example:** `switch# show feature`	(Optional) Displays which features are enabled on the device.

	Command or Action	Purpose
Step 5	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to disable the vPC feature:

```
switch# configure terminal
switch(config)# no feature vpc
switch(config)# exit
switch#
```

Creating a vPC Domain and Entering vpc-domain Mode

You can create a vPC domain and put the vPC Peer-Link port channels into the identical vPC domain on both vPC peer devices. Use a unique vPC domain number throughout a single vPC domain . This domain ID is used to automatically to form the vPC system MAC address.

You can also use this command to enter vpc-domain command mode.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id*
3. **exit**
4. **show vpc brief**
5. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id*
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Creates a vPC domain on the device, and enters vpc-domain configuration mode for configuration purposes. There is no default; the range is from 1 to 1000.

	Command or Action	Purpose

Step 3	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 4	**show vpc brief** **Example:** `switch# show vpc brief`	(Optional) Displays brief information about each vPC domain.
Step 5	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to enter the vpc-domain command mode to configure an existing vPC domain:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# exit
switch(config)#
```

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.

Note You must configure the vPC peer-keepalive link before the system can form the vPC Peer-Link.

Note We recommend that you configure a separate VRF instance and put a Layer 3 port from each vPC peer device into that VRF for the vPC peer-keepalive link. Do not use the vPC Peer-Link itself to send vPC peer-keepalive messages. For information about creating and configuring VRFs, see the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide. Ensure that both the source and destination IP addresses use for the peer-keepalive message are unique in your network. The management port and management VRF are the defaults for these keepalive messages.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id*
3. **peer-keepalive destination** *ipaddress* [**hold-timeout** *secs* | **interval** *msecs* {**timeout** *secs*} | {**precedence** {*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**}]
4. **exit**
5. **show vpc statistics**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id*
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Creates a vPC domain on the device, and enters vpc-domain configuration mode.

Step 3	**peer-keepalive destination** *ipaddress* [**hold-timeout** *secs*	**interval** *msecs* {**timeout** *secs*}	{**precedence** {*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**}] **Example:** `switch(config-vpc-domain)# peer-keepalive destination 172.28.230.85` `switch(config-vpc-domain)#`	Configures the IPv4 and IPv6 addresses for the remote end of the vPC peer-keepalive link. **Note** The system does not form the vPC Peer-Link until you configure a vPC peer-keepalive link. **Note** You may get the following error message if you do not specify the source IP address when you configure an IPv6 address for the remote end of the vPC peer-keepalive link. `Cannot configure IPV6 peer-keepalive without source IPV6 address` The management ports and VRF are the defaults. **Note** We recommend that you configure a separate VRF and use a Layer 3 port from each vPC peer device in that VRF for the vPC peer-keepalive link. For more information about creating and configuring VRFs, see the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide.

	Command or Action	Purpose

Step 4	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits global configuration mode.

Step 5	**show vpc statistics** **Example:** `switch# show vpc statistics`	(Optional) Displays information about the configuration for the keepalive messages.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

For more information about configuring VRFs, see the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide.

This example shows how to configure the destination and source IP address and VRF for the vPC-peer-keepalive link:

```
switch# configure terminal
switch(config)# vpc domain 100
switch(config-vpc-domain)# peer-keepalive destination 172.168.1.2 source 172.168.1.1 vrf
vpc-keepalive
switch(config-vpc-domain)# exit
switch#
```

Creating a vPC Peer-Link

You create the vPC Peer-Link by designating the port channel that you want on each device as the vPC Peer-Link for the specified vPC domain. We recommend that you configure the Layer 2 port channels 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 device for redundancy.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-number*
3. **switchport mode trunk**
4. **switchport trunk allowed vlan** *vlan-list*
5. **vpc peer-link**
6. **exit**

7. **show vpc brief**
8. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface port-channel** *channel-number*
	Example:
	`switch(config)# interface port-channel 20`
	`switch(config-if)#`
		Selects the port channel that you want to use as the vPC Peer-Link for this device, and enters interface configuration mode.

Step 3	**switchport mode trunk** **Example:** `switch(config-if)# switchport mode trunk`	(Optional) Configures this interface in trunk mode.
Step 4	**switchport trunk allowed vlan** *vlan-list* **Example:** `switch(config-if)# switchport trunk allowed vlan 1-120,201-3967`	(Optional) Configures the permitted VLAN list.

Step 5	**vpc peer-link**
	Example:
	`switch(config-if)# vpc peer-link`
	`switch(config-vpc-domain)#`
		Configures the selected port channel as the vPC Peer-Link, and enters vpc-domain configuration mode.

Step 6	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 7	**show vpc brief** **Example:** `switch# show vpc brief`	(Optional) Displays information about each vPC, including information about the vPC Peer-Link.
Step 8	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure a vPC Peer-Link:

```
switch# configure terminal
switch(config)# interface port-channel 20
switch(config-if)# switchport mode
switch(config-if)# switchport mode trunk
switch(config-if)# switchport trunk allowed vlan 1-120,201-3967
switch(config-if)# vpc peer-link
switch(config-vpc-domain)# exit
switch(config)#
```

Moving Other Port Channels into a vPC

We recommend that you attach the vPC domain downstream port channel to two devices for redundancy.

To connect to the downstream device, you create a port channel from the downstream device to the primary vPC peer device and you create another port channel from the downstream device to the secondary peer device. On each vPC peer device, you assign a vPC number to the port channel that connects to the downstream device. You will experience minimal traffic disruption when you are creating vPCs.

Before you begin

Ensure that you have enabled the vPC feature.

Ensure that you are using a Layer 2 port channel.

SUMMARY STEPS

1. **configure terminal**
2. **interface port-channel** *channel-number*
3. **vpc** *number*
4. **exit**
5. **show vpc brief**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

	Command or Action	Purpose

Step 2	**interface port-channel** *channel-number*
	Example:
	`switch(config)# interface port-channel 20`
	`switch(config-if)#`
		Selects the port channel that you want to put into the vPC to connect to the downstream device, and enters interface configuration mode.

Step 3	**vpc** *number*
	Example:
	`switch(config-if)# vpc 5`
	`switch(config-vpc-domain)#`
		Configures the selected port channel into the vPC to connect to the downstream device. You can use any module in the device for these port channels. The range is from 1 and 4096.
		Note
		The vPC number that you assign to the port channel connecting to the downstream device from the vPC peer device must be identical on both vPC peer devices.

Step 4	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc brief** **Example:** `switch# show vpc brief`	(Optional) Displays information on the vPCs.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure a port channel to connect to the downstream device:

```
switch# configure terminal
switch(config)# interface port-channel 20
switch(config-if)# vpc 5
switch(config-if)# exit
switch(config)#
```

Checking the Configuration Compatibility on a vPC Peer-Link

After you have configured the vPC Peer-Link on both vPC peer devices, check that the configurations are consistent on all vPC interfaces. See the “Compatibility Parameters for vPC Interfaces” section for information about consistent configurations on the vPCs.

SUMMARY STEPS

1. **configure terminal**

2. **show vpc consistency-parameters** {**global** | **interface port-channel** *channel-number*}

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**show vpc consistency-parameters** {**global** | **interface port-channel** *channel-number*}
	Example:
	`switch(config)# show vpc consistency-parameters global`
	`switch(config)#`
		(Optional) Displays the status of those parameters that must be consistent across all vPC interfaces.

Example

This example shows how to check that the required configurations are compatible across all the vPC interfaces:

```
switch# configure terminal
switch(config)# show vpc consistency-parameters global
switch(config)#
```

Note Messages regarding the vPC interface configuration compatibility are also logged to the syslog.

Configuring a Graceful Consistency Check

You can configure the graceful consistency check feature, which is enabled by default. Unless this feature is enabled, the vPC is completely suspended when a mismatch in a mandatory compatibility parameter is introduced in a working vPC. When this feature is enabled, only the links on the secondary peer device are suspended. See the “Compatibility Parameters for vPC Interfaces” section for information about consistent configurations on the vPCs.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id*
3. **graceful consistency-check**
4. **exit**
5. **show vpc brief**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id*
	Example:
	`switch(config-if)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Creates a vPC domain if it does not already exist, and enters vpc-domain configuration mode.

Step 3	**graceful consistency-check** **Example:** `switch(config-vpc-domain)# graceful consistency-check`	Specifies that only the links on the secondary peer device are suspended when a mismatch is detected in a mandatory compatibility parameter. Use the **no** form of this command to disable the feature.

Step 4	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc brief** **Example:** `switch# show vpc brief`	(Optional) Displays information on the vPCs.

Example

This example shows how to enable the graceful consistency check feature:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# graceful consistency-check
switch(config-vpc-domain)# exit
switch(config)#
```

Configuring a vPC Peer-Gateway

You can configure vPC peer devices to act as the gateway for packets that are destined to the vPC peer device's MAC address.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id*
3. **peer-gateway**
4. **exit**
5. **show vpc brief**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id*
	Example:
	`switch(config-if)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Creates a vPC domain if it does not already exist, and enters vpc-domain configuration mode.

Step 3	**peer-gateway** **Example:** `switch(config-vpc-domain)# peer-gateway` **Note** Disable IP redirects on all interface-vlans of this vPC domain for correct operation of this feature.	Enables Layer 3 forwarding for packets destined to the peer's gateway MAC address.

Step 4	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc brief** **Example:** `switch# show vpc brief`	(Optional) Displays information about each vPC, including information about the vPC Peer-Link.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring the vPC Peer Switch

You can configure the Cisco Nexus 9000 Series device to make a pair of vPC devices appear as a single STP root in the Layer 2 topology.

Configuring a Pure vPC Peer Switch Topology

You can configure a pure vPC peer switch topology by using the peer-switch command and then setting the best possible (lowest) spanning tree bridge priority value.

Before you begin

Ensure that you have enabled the vPC feature.

Note When using a non-VPC dedicated trunk link between the VPC peers, the non-VPC VLANs should have a different global priority on the peers to prevent STP from blocking the VLANs.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id*
3. **peer-switch**
4. **spanning-tree vlan** *vlan-range* **priority** *value*
5. **exit**
6. **show spanning-tree summary**
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id*
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Enters the vPC domain number that you want to configure, and enters vpc-domain configuration mode.

Step 3	**peer-switch** **Example:** `switch(config-vpc-domain)# peer-switch`	Enables the vPC switch pair to appear as a single STP root in the Layer 2 topology. Use the **no** form of the command to disable the peer switch vPC topology.

	Command or Action	Purpose
Step 4	**spanning-tree vlan** *vlan-range* **priority** *value* **Example:** `switch(config)# spanning-tree vlan 1 priority 8192`	Configures the bridge priority of the VLAN. Valid values are multiples of 4096. The default value is 32768.

Step 5	**exit**
	Example:
	`switch(config-vpc-domain)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 6	**show spanning-tree summary** **Example:** `switch# show spanning-tree summary`	(Optional) Displays a summary of the spanning tree port states including the vPC peer switch.
Step 7	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure a pure vPC peer switch topology:

```
switch# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
switch(config)# vpc domain 5
switch(config-vpc-domain)# peer-switch

2010 Apr 28 14:44:44 switch %STP-2-VPC_PEERSWITCH_CONFIG_ENABLED: vPC peer-switch
configuration is enabled. Please make sure to configure spanning tree "bridge" priority as
per recommended guidelines to make vPC peer-switch operational.

switch(config-vpc-domain)# spanning-tree vlan 1 priority 8192
switch(config-vpc-domain)# exit
switch(config)#
```

Configuring the Suspension of Orphan Ports

When a device that is not vPC-capable connects to each peer, the connected ports are known as orphan ports because they are not members of a vPC. You can explicitly declare physical interfaces as orphan ports to be suspended (shut down) by the secondary peer when it suspends its vPC ports in response to a vPC Peer-Link or peer-keepalive failure. The orphan ports are restored when the vPC is restored.

Note You can configure vPC orphan port suspension only on physical ports, portchannels. However, you cannot configure the same on individual port channel member ports.

vPC orphan port suspend is not supported under vPC member ports.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **show vpc orphan-ports**
3. **interface** *type slot/port*
4. **vpc orphan-port suspend**
5. **exit**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**show vpc orphan-ports** **Example:** `switch# show vpc orphan-ports`	(Optional) Displays a list of the orphan ports.

Step 3	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 3/1`
	`switch(config-if)#`
		Specifies an interface to configure, and enters interface configuration mode.

Step 4	**vpc orphan-port suspend** **Example:** `switch(config-if)# vpc orphan-ports suspend`	Configures the selected interface as a vPC orphan port to be suspended by the secondary peer in the case of a vPC failure.

Step 5	**exit**
	Example:
	`switch(config-if)# exit`
	`switch#`
		Exits interface configuration mode.

Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to configure an interface as a vPC orphan port to be suspended by the secondary peer in the case of a vPC failure:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# vpc orphan-ports suspend
switch(config-if)# exit
switch(config)#
```

Beginning Cisco NX-OS Release 9.2(1), the output of the **show vpc orphan-ports** command is slightly different from that of the earlier releases. This example shows the output of **show vpc orphan-ports**command:

```
switch# show vpc orphan-ports
--------::Going through port database. Please be patient.::--------
VLAN            Orphan Ports
-------         -------------------------
1               Eth1/18, Eth3/23

2               Eth3/23

3               Eth3/23

4               Eth3/23

5               Eth3/23
```

Configuring vPC Object Tracking Tracking Feature on a Single-Module vPC

If you must configure all the vPC Peer-Links and core-facing interfaces on a single module, you should configure a track object and a track list that is associated with the Layer 3 link to the core and on all the links on the vPC Peer-Link on both primary vPC peer devices. Once you configure this feature and if the primary vPC peer device fails, the system automatically suspends all the vPC links on the primary vPC peer device. This action forces all the vPC traffic to the secondary vPC peer device until the system stabilizes.

You must put this configuration on both vPC peer devices. Additionally, you should put the identical configuration on both vPC peer devices because either device can become the operationally primary vPC peer device.

Before you begin

Ensure that you have enabled the vPC feature.

Ensure that you have configured the track object and the track list. Ensue that you assign all interfaces that connect to the core and to the vPC Peer-Link to the track-list object on both vPC peer devices.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id*
3. **track** *track-object-id*
4. **exit**
5. **show vpc brief**

6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id*
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Enters the vPC domain number that you want to configure, and enters vpc-domain configuration mode.

Step 3	**track** *track-object-id*
	Example:
	`switch(config-vpc-domain)# track object 23`
	`switch(config-vpc-domain)#`
		Adds the previously configured track-list object with its associated interfaces to the vPC domain. See the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide for information about configuring object tracking and track lists.

Step 4	**exit**
	Example:
	`switch(config-vpc-domain)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc brief** **Example:** `switch# show vpc brief`	(Optional) Displays information about the tracked objects.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to put the previously configured track-list object into the vPC domain on the vPC peer device:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# track object 5
switch(config-vpc-domain)# exit
switch(config)#
```

Configuring for Recovery After an Outage

If an outage occurs, the vPC waits for a peer adjacency to form on a switch reload. This situation can result in an unacceptably long service disruption. You can configure the Cisco Nexus 9000 Series device to restore vPC services when its peer fails to come on line.

Configuring an Autorecovery

You can configure the Cisco Nexus 9000 Series device to restore vPC services when its peer fails to come online by using the auto-recovery command.

You can configure the Cisco Nexus 9000 Series device to restore vPC services on the secondary vPC peer when its vPC primary peer fails and bringing down peer-keepalive and vPC Peer-Link, by using the **auto-recovery** command. In case of failure of primary switch where both peer-keepalive and vPC Peer-Links are down secondary switch will suspend vPC member. However, after 3 missed keepalive heartbeats secondary switch resumes the role of a primary switch and bring up vPC member ports. The **auto-recovery reload restore** command can be used in scenarios when vPC primary switch reloads, where secondary switch resumes the role of the vPC primary and bring ip VPC member ports.

Note The auto-recovery feature is not enabled by default on Cisco Nexus 9000 Switches. When the object tracking is triggered, the vPC secondary peer device does not change its role to that primary device and it reinitializes the vPC legs. You must manually configure auto-recovery on the vPC secondary peer device so that it can take over the primary role and reinitialize its vPC legs.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id*
3. **auto-recovery** [**reload-delay** *time*]
4. **exit**
5. **show running-config vpc**
6. **show vpc consistency-parameters interface port-channel** *number*
7. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

	Command or Action	Purpose

Step 2	**vpc domain** *domain-id*
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Enters the vPC domain number that you want to configure, and enters vpc-domain configuration mode.

Step 3	**auto-recovery** [**reload-delay** *time*] **Example:** `switch(config-vpc-domain)# auto-recovery`	Configures the vPC to assume its peer is not functional and to bring up the vPC, and specifies the time to wait after a reload to restore the vPC. The default delay is 240 seconds. You can configure a delay from 240 to 3600 seconds. Use the **no** form of the command to reset the vPC to its default settings.

Step 4	**exit**
	Example:
	`switch(config-vpc-domain)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show running-config vpc** **Example:** `switch# show running-config vpc`	(Optional) Displays information about the vPC, specifically the reload status.
Step 6	**show vpc consistency-parameters interface port-channel** *number* **Example:** `switch# show vpc consistency-parameters interface port-channel 1`	(Optional) Displays information about the vPC consistency parameters for the specified interface.
Step 7	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration. **Note** To ensure the autorecovery feature is enabled, you should perform this step.

Example

This example shows how to set the vPC autorecovery feature and save it in the switch startup configuration:

```
switch# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
switch(config)# vpc domain 5
switch(config-vpc-domain)# auto-recovery
switch(config-vpc-domain)# auto-recovery auto-recovery reload-delay 100

Warning:
Enables restoring of vPCs in a peer-detached state after reload, will wait for 240
seconds to determine if peer is un-reachable
```

```
switch(config-vpc-domain)# exit
switch(config)# exit
switch# copy running-config startup-config
```

Configuring Hitless vPC Role Change

Complete these steps to enable hitless vPC role change.

Before you begin

- Ensure that the vPC feature is enabled.
- Ensure that the vPC Peer-Link is up.
- Verify the role priority of devices.
- Verify the existing configured role priority before configuring vPC hitless role change feature In a vPC domain, enable the **peer-switch** command, where both vPC peers have same STP priorities, and ensure it is operational before issuing a role change. If you do not enable the **peer-switch** command, it can lead to convergence issues.

SUMMARY STEPS

1. **vpc role preempt**
2. **show vpc role**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**vpc role preempt**
	Example:
	`switch# vpc role preempt`
	`switch(config)#`
		Enable hitless vPC role change.

Step 2	**show vpc role** **Example:** `switch(config)# show vpc role`	(Optional) Verify hitless vPC role change feature.

Example

This example on how to configure hitless vPC role change:

```
switch# show vpc rolevPC Role status
----------------------------------------------------
vPC role                        : secondary
vPC system-mac                  : 00:23:04:ee:be:01
vPC system-priority             : 32667
vPC local system-mac            : 8c:60:4f:03:84:41
```

```
vPC local role-priority         : 32668
vPC peer system-mac             : 8c:60:4f:03:84:43
vPC peer role-priority          : 32667

! Configure vPC hitless role change on the device!

switch(config)# vpc role preempt
! The following is an output from the show vpc role command after the
vPC hitless feature is configured
switch(config)# show vpc role
vPC Role status
----------------------------------------------------
vPC role                        : primary
vPC system-mac                  : 00:00:00:00:00:00
vPC system-priority             : 32667
vPC local system-mac            : 8c:60:4f:03:84:41
vPC local role-priority         : 32666
vPC peer system-mac             : 8c:60:4f:03:84:43
vPC peer role-priority          : 32667

switch(config)#
```

Use Case Scenario for vPC Role Change

The hitless vPC role change feature can be used in the following scenarios:

- Role change request—When you want to change the roles of the peer devices in a vPC domain.
- Primary switch reload—When the devices comes up after a reload and roles are defined, you can use the hitless vPC role change feature to restore the roles. For example, after a reload if the primary device takes the role of operational secondary and the secondary device takes the role of primary operational, you can change the vPC peer roles to their original defined roles using the **vpc role preempt** command.

Note Always check the existing device role priority before switching vPC role.

- Dual-active recovery—In a dual-active recovery scenario, the vPC primary switch continues to be (operational) primary, but the vPC secondary switch becomes the targeted primary switch and keeps its vPC member ports up. You can use the vPC hitless feature and restore the device roles. After the Dual-active recovery, if one side is operational primary and the other side operational secondary, then you can use the **vpc role preempt** command to restore the device roles to be primary and secondary

Manually Configuring a vPC Domain MAC Address

When you create a vPC domain, the Cisco NX-OS software automatically creates a vPC system MAC address, which is used for operations that are confined to the link-scope, such as LACP. However, you might choose to configure the vPC domain MAC address manually.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id*
3. **system-mac** *mac-address*
4. **exit**
5. **show vpc role**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id*
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Enters the vPC domain number that you want to configure. The system enters vpc-domain configuration mode.

Step 3	**system-mac** *mac-address*
	Example:
	`switch(config-vpc-domain)# system-mac 23fb.4ab5.4c4e`
	`switch(config-vpc-domain)#`
		Enters the MAC address that you want for the specified vPC domain in the following format: aaaa.bbbb.cccc.

Step 4	**exit**
	Example:
	`switch(config-vpc-domain)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc role** **Example:** `switch# show vpc brief`	(Optional) Displays the vPC system MAC address.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to manually configure a vPC domain MAC address:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# system-mac 13gb.4ab5.4c4e
switch(config-vpc-domain)# exit
switch(config)#
```

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.

Note We recommend that you manually configure the vPC system priority when you are running LACP to ensure that the vPC peer devices are the primary devices on LACP. When you manually configure the system priority, ensure that you configure the same priority value on both vPC peer devices. If these values do not match, vPC does not come up.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id*
3. **system-priority** *priority*
4. **exit**
5. **show vpc role**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id*
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Enters the vPC domain number that you want to configure. The system enters vpc-domain configuration mode.

	Command or Action	Purpose

Step 3	**system-priority** *priority*
	Example:
	`switch(config-vpc-domain)# system-priority 4000`
	`switch(config-vpc-domain)#`
		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.

Step 4	**exit**
	Example:
	`switch(config-vpc-domain)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc role** **Example:** `switch# show vpc role`	(Optional) Displays the vPC system priority.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to manually configure the vPC domain system priority:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# system-priority 4000
switch(config-vpc-domain)# exit
switch(config)#
```

Manually Configuring the vPC Peer Device Role

By default, the Cisco NX-OS software elects a primary and secondary vPC peer device after you configure the vPC domain and both sides of the vPC Peer-Link. However, you might want to elect a specific vPC peer device as the primary device for the vPC. Then, you would manually configure the role value for the vPC peer device that you want as the primary device to be lower than the other vPC peer device.

vPCs do not support role preemption. If the primary vPC peer device fails, the secondary vPC peer device takes over to become operationally the vPC primary device. However, the original operational roles are not restored if the formerly primary vPC comes up again.

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id*
3. **role priority** *priority*

4. **exit**
5. **show vpc role**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**vpc domain** *domain-id*
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Enters the vPC domain number that you want to configure. The system enters vpc-domain configuration mode.

Step 3	**role priority** *priority*
	Example:
	`switch(config-vpc-domain)# role priority 4`
	`switch(config-vpc-domain)#`
		Enters the role priority that you want for the vPC system priority. The range of values is from 1 to 65636, and the default value is 32667. A lower value means that this switch has a better chance of being the primary vPC.

Step 4	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Exits vpc-domain configuration mode.

Step 5	**show vpc role** **Example:** `switch# show vpc role`	(Optional) Displays the vPC system priority.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Example

This example shows how to manually configure the role priority of the vPC peer device:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# role priority 4
switch(config-vpc-domain)# exit
switch(config)#
```

Enabling STP to Use the Cisco MAC Address

This procedure enables STP to use the Cisco MAC address (00:26:0b:xx:xx:xx).

Before you begin

Ensure that you have enabled the vPC feature.

SUMMARY STEPS

1. **configure terminal**
2. **vpc domain** *domain-id*
3. **[no] mac-address bpdu source version 2**
4. **exit**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**vpc domain** *domain-id* **Example:** `switch(config)# vpc domain 5`	Creates a vPC domain if it does not already exist, and enters vpc-domain configuration mode.
Step 3	**[no] mac-address bpdu source version 2** **Example:** `switch(config-vpc-domain)# mac-address bpdu source version 2`	Enables STP to use the Cisco MAC address (00:26:0b:xx:xx:xx) as the source address of BPDUs generated on vPC ports.
Step 4	**exit** **Example:** `switch(config-vpc-domain)# exit`	Exits vpc-domain configuration mode.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(confg)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Verifying the vPC Configuration

To display vPC configuration information, perform one of the following tasks:

Command	Purpose
show feature	Displays whether the vPC is enabled or not.
show vpc brief	Displays brief information about the vPCs.
show vpc consistency-parameters	Displays the status of those parameters that must be consistent across all vPC interfaces.
show running-config vpc	Displays running configuration information for vPCs.
show port-channel capacity	Displays how many port channels are configured and how many are still available on the device.
show vpc statistics	Displays statistics about the vPCs.
show vpc peer-keepalive	Displays information about the peer-keepalive messages.
show vpc role	Displays the peer status, the role of the local device, the vPC system MAC address and system priority, and the MAC address and priority for the local vPC device.

Viewing Dual Active Detection Status

When the vPC Peer-Link is down but the peer-keepalive remains up, the vPC secondary switch shuts down all of its vPC member ports. In this scenario, the dual active detection status is set to 1 on the operational secondary device to indicate that its member ports are shut down. The dual active detection status remains 0 on the operational primary device.

This example displays dual active detection status on operational secondary device:

```
switch# show vpc role
vPC Role status
----------------------------------------------------
vPC role                        :primary, operational secondary
Dual Active Detection Status    : 1
vPC system-mac                  : 00:23:04:ee:be:01
vPC system-priority             : 32667
vPC local system-mac            : 24:6c:84:34:c8:77
vPC local role-priority         : 200
vPC local config role-priority  : 200
vPC peer system-mac             : 24:6c:84:34:bf:df
vPC peer role-priority          : 300
vPC peer config role-priority   : 300
switch#
```

This example displays the dual active detection status on operational primary device:

```
switch# show vpc role
vPC Role status
----------------------------------------------------
vPC role                        :secondary, operational primary
Dual Active Detection Status    : 0
vPC system-mac                  : 00:23:04:ee:be:01
vPC system-priority             : 32667
vPC local system-mac            : 24:6c:84:34:bf:df
vPC local role-priority         : 300
```

```
vPC local config role-priority  : 300
vPC peer system-mac             : 24:6c:84:34:c8:77
vPC peer role-priority          : 200
vPC peer config role-priority   : 200
switch#
```

Monitoring vPCs

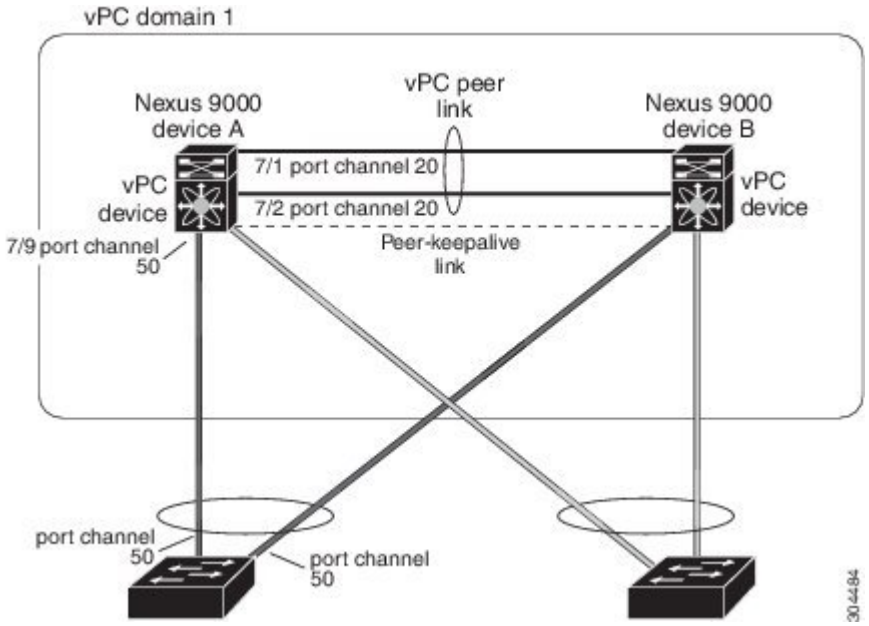
Use the **show vpc statistics**command to display vPC statistics.

Note This command displays the vPC statistics only for the vPC peer device that you are working on.

Configuration Examples for vPCs

The following example shows how to configure vPC on device A as shown in the figure:

Figure 28: vPC Configuration Example



1. Enable vPC and LACP.

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

2. (Optional) Configure one of the interfaces that you want to be a vPC Peer-Link in the dedicated port mode.

```
switch(config)# interface ethernet 7/1,
ethernet 7/3, ethernet 7/5. ethernet 7/7
switch(config-if)# shutdown
```

```
switch(config-if)# exit
switch(config)# interface ethernet 7/1

switch(config-if)# no shutdown
switch(config-if)# exit
switch(config)#
```

3. (Optional) Configure the second, redundant interface that you want to be a vPC Peer-Link in the dedicated port mode.

```
switch(config)# interface ethernet 7/2, ethernet 7/4,
ethernet 7/6. ethernet 7/8
switch(config-if)# shutdown
switch(config-if)# exit
switch(config)# interface ethernet 7/2

switch(config-if)# no shutdown
switch(config-if)# exit
switch(config)#
```

4. Configure the two interfaces (for redundancy) that you want to be in the vPC Peer-Link to be an active Layer 2 LACP port channel.

```
switch(config)# interface ethernet 7/1-2
switch(config-if)# switchport
switch(config-if)# switchport mode trunk
switch(config-if)# switchport trunk allowed vlan 1-50
switch(config-if)# switchport trunk native vlan 20
switch(config-if)# channel-group 20 mode active
switch(config-if)# exit
```

5. Create and enable the VLANs.

```
switch(config)# vlan 1-50
switch(config-vlan)# no shutdown
switch(config-vlan)# exit
```

6. Create a separate VRF for the vPC peer-keepalive link and add a Layer 3 interface to that VRF.

```
switch(config)# vrf context pkal
switch(config-vrf)# exit
switch(config)# interface ethernet 8/1
switch(config-if)# vrf member pkal
switch(config-if)# ip address 172.23.145.218/24
switch(config-if)# no shutdown
switch(config-if)# exit
```

7. Create the vPC domain and add the vPC peer-keepalive link.

```
switch(config)# vpc domain 1
switch(config-vpc-domain)# peer-keepalive
 destination 172.23.145.217 source 172.23.145.218 vrf pkal
switch(config-vpc-domain)# exit
```

8. Configure the vPC vPC Peer-Link.

```
switch(config)# interface port-channel 20
switch(config-if)# switchport mode trunk
switch(config-if)# switchport trunk allowed vlan 1-50
switch(config-if)# vpc peer-link
switch(config-if)# exit
switch(config)#
```

9. Configure the interface for the port channel to the downstream device of the vPC.

```
switch(config)# interface ethernet 7/9
switch(config-if)# switchport mode trunk
switch(config-if)# allowed vlan 1-50
switch(config-if)# native vlan 20
switch(config-if)# channel-group 50 mode active
switch(config-if)# exit
switch(config)# interface port-channel 50
switch(config-if)# vpc 50
switch(config-if)# exit
switch(config)#
```

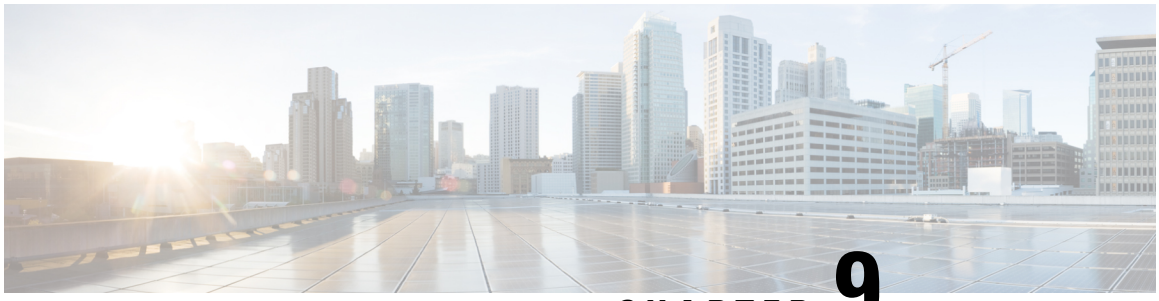
10. Save the configuration.

```
switch(config)# copy running-config startup-config
```

Note If you configure the port channel first, ensure that it is a Layer 2 port channel.

Related Documents

Related Topic	Related Topic
System management	System management
High availability	High availability
Release Notes	Release Notes



CHAPTER 9

Configuring IP Tunnels

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 Tunnel Overview

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 GRE as a carrier protocol.
- 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.

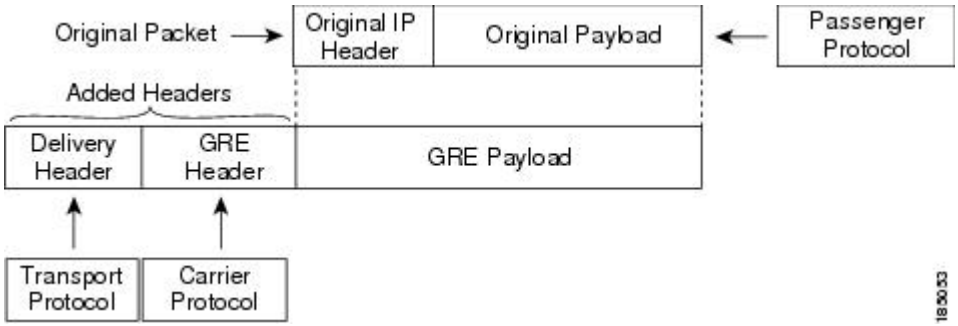
You must enable the tunnel feature before you can configure it. The system automatically takes a checkpoint prior to disabling the feature, and you can roll back to this checkpoint. See the Cisco Nexus 9000 Series NX-OS System Management Configuration Guide for information about rollbacks and checkpoints.

GRE Tunnels

You can use generic routing encapsulation (GRE) as the carrier protocol for a variety of passenger protocols.

The following 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 29: GRE PDU



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

The 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. This type of tunnel will carry both inbound and outbound traffic.

Note Beginning with Cisco NX-OS Release 10.3(3)F, the selection of GRE or IP-in-IP tunnel destination based on the PBR policy is supported.

Note IP-in-IP tunnel encapsulation and decapsulation is not supported on Cisco Nexus 9500 Series switches with N9K-X9636C-R, N9K-X9636Q-R, N9K-X9636C-RX line cards.

Note IP-in-IP tunnel encapsulation and decapsulation is not supported on a vPC setup on Cisco Nexus 9300-EX, 9300-FX, 9300-GX and Nexus 9500 platform switches.

Multi-Point IP-in-IP Tunnel Decapsulation

The 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.

Path MTU Discovery

Path maximum transmission unit (MTU) discovery (PMTUD) prevents fragmentation in the path between two endpoints by dynamically determining the lowest MTU along the path from the packet's source to its destination. PMTUD reduces the send MTU value for the connection if the interface receives information that the packet would require fragmentation.

When you enable PMTUD, the interface sets the Don't Fragment (DF) bit on all packets that traverse the tunnel. If a packet that enters the tunnel encounters a link with a smaller MTU than the MTU value for the packet, the remote link drops the packet and sends an ICMP message back to the sender of the packet. This message indicates that fragmentation was required (but not permitted) and provides the MTU of the link that dropped the packet.

Note PMTUD on a tunnel interface requires that the tunnel endpoint can receive ICMP messages generated by devices in the path of the tunnel. Check that ICMP messages can be received before using PMTUD over firewall connections.

High Availability

IP tunnels support stateful restarts. A stateful restart occurs on a supervisor switchover. After the switchover, Cisco NX-OS applies the runtime configuration after the switchover.

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 must enable the tunneling feature in a device before you can configure and enable any IP tunnels.

Guidelines and Limitations

IP tunnels have the following configuration guidelines and limitations:

- Beginning with Cisco NX-OS Release 9.3(3):
  - Total number of 16 GRE/IPIP tunnels are supported on Cisco Nexus 9200, 9300-EX/FX/FX2 switches, and 9500 switches with 9700-EX/FX line cards.
  - More than 1 and up to 16 IPIP Decap-any tunnels are supported - 1 decap-any tunnel per VRF. This is supported on Cisco Nexus 9200, and 9300-EX/FX/FX2 platforms.
  - VRF membership of the interface, where IPIP/GRE encapsulated packets are ingressing on the terminating node, should match with the tunnel transport VRF for tunnel to correctly terminate the packets.

    - The IPIP/GRE packet coming on a non default VRF may get terminated by a tunnel in default VRF if the packet outer header matches with the tunnel source and the tunnel destination.

- Beginning with Cisco NX-OS Release 9.3(5), the following features are supported on N9K-C9316D-GX, N9K-C93600CD-GX and N9K-C9364C-GX switches:
    - A total number of 16 GRE/IPIP tunnels.
    - More than 1 and upto 16 IPIP Decap-any tunnels are supported -- 1 decap-any tunnel per VRF.
- You must configure multiple GRE or IP-in-IP tunnels that use the same outer transport VRF (**tunnel use-vrf**) with a unique tunnel destination IP, per tunnel in these platforms:
    - N9K-X9736C-FX, N9K-X9736Q-FX, N9K-X9788TC-FX, N9K-C93180YC-FXN9K-C93108TC-FX, N9K-C9348GC-F, N9K-C9348GC-FXP, N9K-C9358GY-FXP, N9K-X9732C-FX, N9K-C92348GC-X
    - N9K-C9336C-FX2-E, N9K-C93216TC-FX2, N9K-C93360YC-FX2, N9K-C93240YC-FX2-Z, N9K-C93240YC-FX2,N9K-C9336C-FX2
    - N9K-C9316D-GX, N9K-C93600CD-GX, N9K-C9364C-GX, N9K-X9716D-GX,
    - N9K-X9736C-FX3, N9K-C93180YC-FX3S, N9K-C93180YC-FX3, N9K-C93108TC-FX3P,N9K-C9348GC-FX3, N9K-C9348GC-FX3PH, N9K-C93108TC-FX3, N9K-C92348GC-FX3
    - N9K-C9364D-GX2A, N9K-C9332D-GX2B, N9K-C9348D-GX2A, N9K-C9408
    - N9K-C9332D-H2R, N9K-C9364C-H1, N9K-C93400LD-H1
- On all Nexus platforms, you must configure multiple GRE or IP-in-IP tunnels that use the same outer transport VRF (**tunnel use-vrf**) with unique tunnel source IP and tunnel destination IP, per tunnel.
- Nexus 9000 switches do not support the coexistence of IP tunnels with FC/FCOE traffic. Bringing up an IP tunnel on a switch with FC/FCOE traffic results in that traffic being dropped.
- The **show** commands with the **internal** keyword are not supported.
- Cisco NX-OS supports only the following protocols:
    - IPv4 passenger protocol.
    - GRE carrier protocol.
- Beginning with Cisco NX-OS Release 9.3(3), the maximum number of supported GRE and IP-in-IP regular tunnels is 16.
- IP tunnels do not support access control lists (ACLs) or QoS policies.
- Cisco NX-OS supports the GRE header defined in IETF RFC 2784. Cisco NX-OS does not support tunnel keys and other options from IETF RFC 1701.
- Cisco NX-OS does not support GRE tunnel keepalives.
- All unicast routing protocols are supported by IP tunnels.
- The IP tunnel interface cannot be configured to be a span source or destination.

- Beginning with Cisco NX-OS Release 10.3(3)F, the selection of GRE or IP-in-IP tunnel destination based on the PBR policy is supported.
- BGP adjacency over tunnel is not supported in a scenario where the tunnel interface and tunnel source are in same VRF (example: VRF-A) and tunnel destination is reachable with route-leak from opposite end (example: via VRF-B)
- GRE tunnels does not support RACLs.
- When setting up a GREv6 or IP-in-IP tunnel, you cannot use different VRFs for the tunnel interface and the tunnel destination. Both must use the same VRF for the tunnel to work properly.You need to use the same VRF for the tunnel interface and the tunnel destination.

  For GREv4, configuring tunnel interface VRF member that is different from the tunnel use-vrf is supported.

```
switch# interface tunnel X
vrf member INNER-VRF
tunnel use-vrf TRANSPORT-VRF
```

- GRE tunnels supports only limited traffic (ingress or egress) counters.
- Layer 3 FEX interfaces not are allowed as tunnel source and/or destination
- Double encapsulation is not allowed on GRE tunnels.
- BFD is not supported on GRE tunnels.
- On Cisco Nexus N9K-C9300-GX platforms, GRE/IPinIP tunnel interfaces cannot co-exist with Dot1Q tagged L2 bcast or 1Q tagged L2/L3 mcast transit traffic. When you configure **feature tunnel** on Cisco Nexus N9300-GX platform, the following warning is displayed and you get a syslog message warning you. You should not configure **feature tunnel** if you have Dot1Q tagged L2 bcast or 1Q tagged L2/L3 mcast transit traffic on the device.

```
N9300-GX(config)# feature tunnel
WARN:GRE/IPinIP cannot coexist with 1Q tagged L2 bcast or 1Q tagged L2/L3 mcast transit
 packets on this
platform
N9300-GX(config)#
N9300-GX(config)# show logging logfile
2019 Dec 12 00:41:08 N9300-GX %TUNNEL-2-TRAFFIC_WARNING: GRE/IPinIP cannot coexist with
 1Q
tagged L2 bcast or 1Q tagged L2/L3 mcast transit packets on this platform
N9300-GX(config)#
```

- The feature **feature tunnel** on the Cisco Nexus 9000 switches cannot co-exist with the VXLAN feature, **feature nv overlay**.
- Cisco Nexus 9200, 9300-EX, 9300-FX, 9300-FX2 series switches and Cisco Nexus 9500 platform switches with 9700-EX/FX line cards may not have multiple tunnel interfaces in a single VRF that are sourced from or destined to the same IP address. For example, a device may not have tunnel 0 and tunnel 1 interfaces in the default VRF that are sourced from the same IP address or interface.
- Cisco Nexus 9300-EX, 9300-FX, 9300-GX and Nexus 9500 platform switches in vPC can act as GRE Tunnel endpoints for their respective tunnels. However, the tunnel destination can not be through a vPC.
- Beginning with Cisco NX-OS Release 10.3(3)F, the PBR policy on a tunnel interface is supported only for **gre ip**, **ipip ip**, and **ipip decapsulate-any ip** modes on Cisco Nexus 9300-FX2/FX3/GX/GX2 platform switches .

- IP tunnels are not supported on Cisco Nexus 9300-FX or Cisco Nexus 9300-FX2 switches if FC or FCOE is configured.

Default Settings

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

Table 19: Default IP Tunnel Parameters

Parameters	Default
Path MTU discovery age timer	10 minutes
Path MTU discovery minimum MTU	64
Tunnel feature	Disabled

Configuring IP Tunnels

Note If you are familiar with the Cisco IOS CLI, be aware that the Cisco NX-OS commands for this feature might differ from the Cisco IOS commands that you would use.

Enabling Tunneling

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

SUMMARY STEPS

1. **configure terminal**
2. **feature tunnel**
3. **exit**
4. **show feature**
5. **copy running-config startup-config**

DETAILED STEPS

Procedure

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

	Command or Action	Purpose
	`switch# configure terminal` `switch(config)#`	

Step 2	**feature tunnel**
	Example:
	`switch(config)# feature tunnel`
	`switch(config-if)#`
		Allows the creation of a new tunnel interface.
		To disable the tunnel interface feature, use the **no** form of this command.
		Note
		The **feature tunnel** command may break the multicast functionality if multicast heavy template is enforced.

Step 3	**exit**
	Example:
	`switch(config-if)# exit`
	`switch#`
		Exits the interface mode and returns to the configuration mode.

Step 4	**show feature** **Example:** `switch(config-if)# show feature`	(Optional) Displays information about the features enabled on the device.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

Creating a Tunnel Interface

You can create a tunnel interface and then configure this logical interface for your IP tunnel.

Note Cisco NX-OS supports a maximum of 8 IP tunnels.

Note Use the **no interface tunnel** command to remove the tunnel interface and all associated configuration.

Command	Purpose
no interface tunnel *number* **Example:** `switch(config)# no interface tunnel 1`	Deletes the tunnel interface and the associated configuration.
description *string* **Example:** `switch(config-if)# description GRE tunnel`	Configures a description for the tunnel.
mtu *value* **Example:** `switch(config-if)# mtu 1400`	Sets the MTU of IP packets sent on an interface.
tunnel ttl *value* **Example:** `switch(config-if)# tunnel ttl 100`	Sets the tunnel time-to-live value. The range is from 1 to 255.

Note Configuring an GREv6 or IP-in-IP tunnel that uses a tunnel interface VRF that is different from the **use-vrf** for the tunnel destination is not supported. You need to use the same VRF for a tunnel interface and the tunnel destination. For GREv4, configuring tunnel interface VRF that is different from the use-vrf for tunnel is supported.

Before you begin

You can configure the tunnel source and the tunnel destination in different VRFs. Ensure that you have enabled the tunneling feature.

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel** *number*
3. **tunnel mode** {**gre ip** | **ipip** {**ip** | **decapsulate-any**}}
4. **tunnel source** {*ip-address* |*interface-name*}
5. **tunnel destination** {*ip-address* |*host-name*}
6. **tunnel use-vrf** *vrf-name*
7. **show interfaces tunnel** *number*
8. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface tunnel** *number*
	Example:
	`switch(config)# interface tunnel 1`
	`switch(config-if)#`
		Creates a new tunnel interface.

Step 3	**tunnel mode** {**gre ip** | **ipip** {**ip** | **decapsulate-any**}}	Sets this tunnel mode to GRE, ipip, or ipip decapsulate-only. The **gre** and **ip** keywords specify that GRE encapsulation over IP will be used. 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	**tunnel source** {*ip-address* |*interface-name*} **Example:** `switch(config-if)# tunnel source ethernet 1/2`	Configures the source address for this IP tunnel. The source can be specified by IP address or logical interface name.
Step 5	**tunnel destination** {*ip-address* |*host-name*} **Example:** `switch(config-if)# tunnel destination 192.0.2.1`	Configures the destination address for this IP tunnel. The destination can be specified by IP address or logical host name.
Step 6	**tunnel use-vrf** *vrf-name* **Example:** `switch(config-if)# tunnel use-vrf blue`	(Optional) Uses the configured VRF to look up the tunnel IP destination address.
Step 7	**show interfaces tunnel** *number* **Example:** `switch# show interfaces tunnel 1`	(Optional) Displays the tunnel interface statistics.
Step 8	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

Example

This example shows how to create a tunnel interface

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

Configuring a Tunnel Interface

You can set a tunnel interface to GRE tunnel mode, ipip mode, or ipip decapsulate-only mode. GRE mode is the default tunnel mode. .

The **tunnel source direct** and **tunnel mode ipv6ipv6 decapsulate-any** CLI commands are supported on Cisco Nexus 9000 Series switches.

Before you begin

Ensure that you have enabled the tunneling feature.

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel** *number*
3. **tunnel mode** {**gre ip** | **ipip** | {**ip** | **decapsulate-any**}}
4. **show interfaces tunnel** *number*
5. **mtu** *value*
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface tunnel** *number*
	Example:
	`switch(config)# interface tunnel 1`
	`switch(config-if)#`
		Creates a new tunnel interface.

Step 3	**tunnel mode** {**gre ip** | **ipip** | {**ip** | **decapsulate-any**}}	Sets this tunnel mode to GRE, ipip, or ipip decapsulate-only.

	Command or Action	Purpose
		The **gre** and **ip** keywords specify that GRE encapsulation over IP will be used. 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	**show interfaces tunnel** *number* **Example:** `switch(config-if)# show interfaces tunnel 1`	(Optional) Displays the tunnel interface statistics.
Step 5	**mtu** *value*	Sets the maximum transmission unit (MTU) of IP packets sent on an interface. The range is from 64 to 9192 units.
Step 6	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

Example

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)# copy running-config startup-config
```

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
```

Configuring a GRE Tunnel

You can set a tunnel interface to GRE tunnel mode.

Note Cisco NX-OS supports only the GRE protocol for IPV4 over IPV4.

Before you begin

Ensure that you have enabled the tunneling feature.

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel** *number*
3. **tunnel mode gre ip**
4. **show interfaces tunnel** *number*
5. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**interface tunnel** *number*
	Example:
	`switch(config)# interface tunnel 1`
	`switch(config-if)#`
		Creates a new tunnel interface.

Step 3	**tunnel mode gre ip** **Example:** `switch(config-if)# tunnel mode gre ip`	Sets this tunnel mode to GRE.
Step 4	**show interfaces tunnel** *number* **Example:** `switch(config-if)# show interfaces tunnel 1`	(Optional) Displays the tunnel interface statistics.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

Enabling Path MTU Discovery

Use the **tunnel path-mtu discovery** command to enable path MTU discovery on a tunnel.

SUMMARY STEPS

1. **tunnel path-mtu-discovery age-timer** *min*

2. **tunnel path-mtu-discovery min-mtu** *bytes*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**tunnel path-mtu-discovery age-timer** *min* **Example:** `switch(config-if)# tunnel path-mtu-discovery age-timer 25`	Enables Path MTU Discovery (PMTUD) on a tunnel interface. • min—Number of minutes. The range is from 10 to 30. The default is 10.
Step 2	**tunnel path-mtu-discovery min-mtu** *bytes* **Example:** `switch(config-if)# tunnel path-mtu-discovery min-mtu 1500`	Enables Path MTU Discovery (PMTUD) on a tunnel interface. • bytes—Minimum MTU recognized. The range is from 64 to 9192. The default is 64.

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.

SUMMARY STEPS

1. **configure terminal**
2. **interface tunnel** *number*
3. **vrf member** *vrf-name*
4. **ip address** *ip-prefix/length*
5. **show vrf** [*vrf-name*] **interface** *interface-type number*
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

	Command or Action	Purpose

Step 2	**interface tunnel** *number*
	Example:
	`switch(config)# interface tunnel 0`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**vrf member** *vrf-name* **Example:** `switch(config-if)# vrf member RemoteOfficeVRF`	Adds this interface to a VRF.
Step 4	**ip address** *ip-prefix/length* **Example:** `switch(config-if)# ip address 192.0.2.1/16`	Configures an IP address for this interface. You must do this step after you assign this interface to a VRF.
Step 5	**show vrf** [*vrf-name*] **interface** *interface-type number* **Example:** `switch(config-vrf)# show vrf Enterprise interface tunnel 0`	(Optional) Displays VRF information.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Saves this configuration change.

Example

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

To verify the IP tunnel configuration information, perform one of the following tasks:

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.

Command	Purpose
show interface tunnel *number* **counters**	Displays interface counters of input/output packets. **Note** The byte count displayed with the interface counters include the internal header size.
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

The following 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 2/1 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
tunnel path-mtu-discovery 25 1500

interface ethernet 1/2
ip address 192.0.2.55/8
```

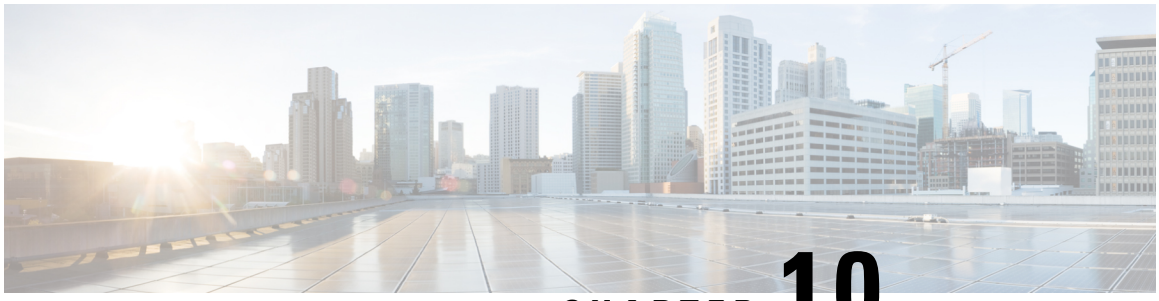
Router B:

```
feature tunnel
interface tunnel 0
ip address 209.165.20.1/8
tunnel source ethernet 2/1
tunnel destination 192.0.2.55
tunnel mode gre ip

interface ethernet 2/1
ip address 192.0.2.2/8
```

Related Documents

Related Topic	Document Title
IP Tunnel commands	*Cisco Nexus 9000 Series NX-OS Interfaces Command Reference*



CHAPTER 10

Configuring Q-in-Q VLAN Tunnels

Information About Q-in-Q Tunnels

This chapter describes how to configure IEEE 802.1Q-in-Q VLAN tunnels and Layer 2 protocol tunneling on Cisco NX-OS devices.

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.

Q-in-Q Tunneling

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 the 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.

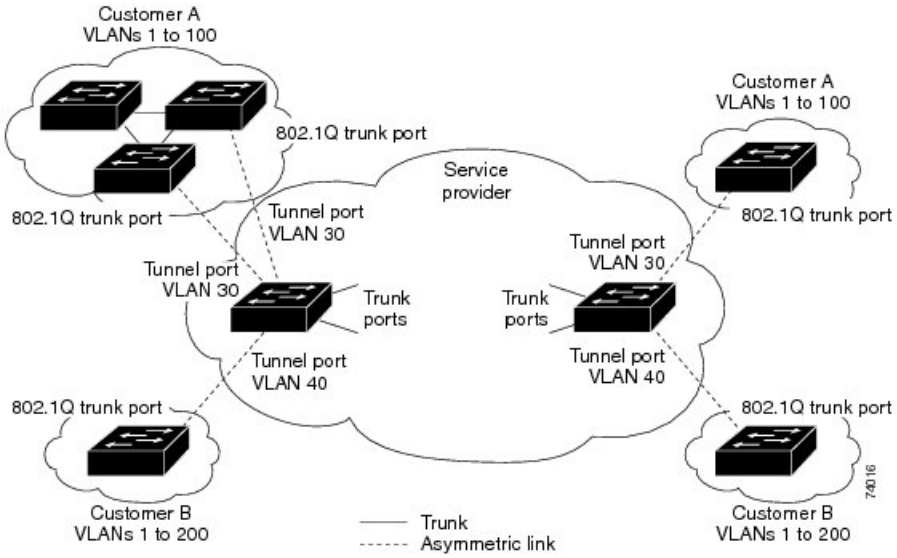
Note Q-in-Q is supported on port channels. To configure a port channel as an asymmetrical link, all ports in the port channel must have the same tunneling configuration.

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 the 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 the 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 that is 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. See the figure below.

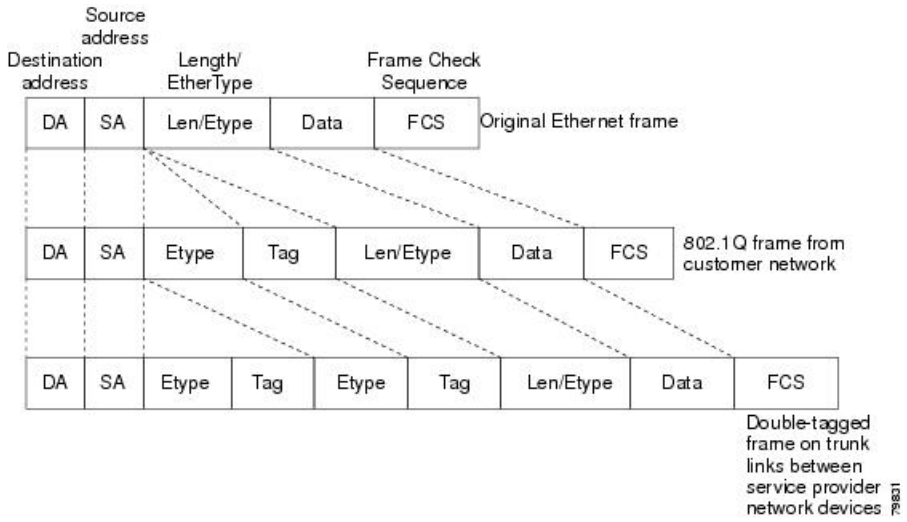
Figure 30: 802.1Q-in-Q Tunnel Ports



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 as shown in the figure below.

Figure 31: 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.

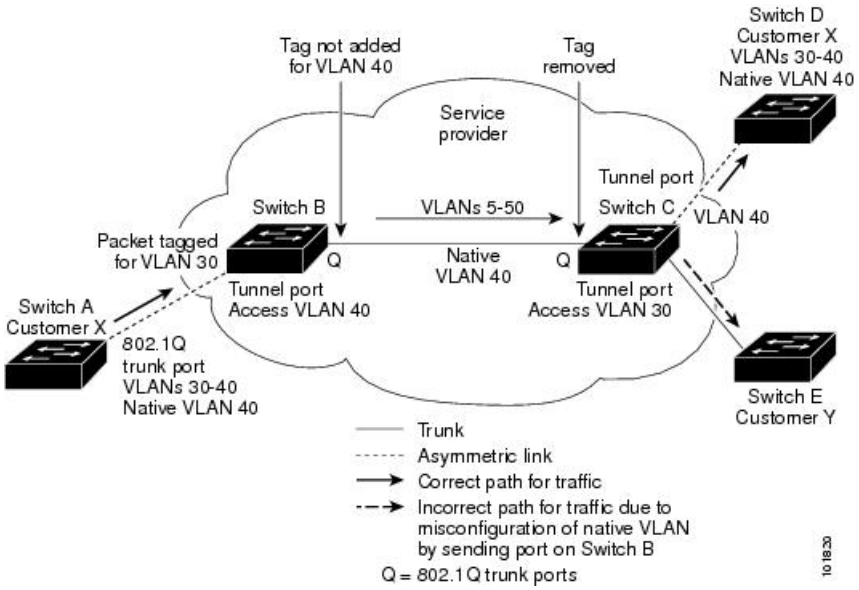
Note Hierarchical tagging, or multi-level dot1q tagging Q-in-Q, is not supported.

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.

In the figure below, 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 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.

Figure 32: Native VLAN Hazard



These are a couple ways to solve the native VLAN problem:

- 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. The 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.

You can configure the switch to allow multi-tagged BPDUs on a tunnel port. If you enable the **l2protocol tunnel allow-double-tag** command, when a multi-tagged customer BPDU enters the tunnel port, the original 802.1Q tags from the customer traffic is preserved and an outer VLAN tag (customer's access VLAN ID, as assigned by the service-provider) is added in the encapsulated packet. Therefore, BPDU packets that enter the service-provider infrastructure are multi tagged. When the BPDUs leave the service-provider network, the outer tag is removed and the original multi-tagged BPDU is sent to the customer network.

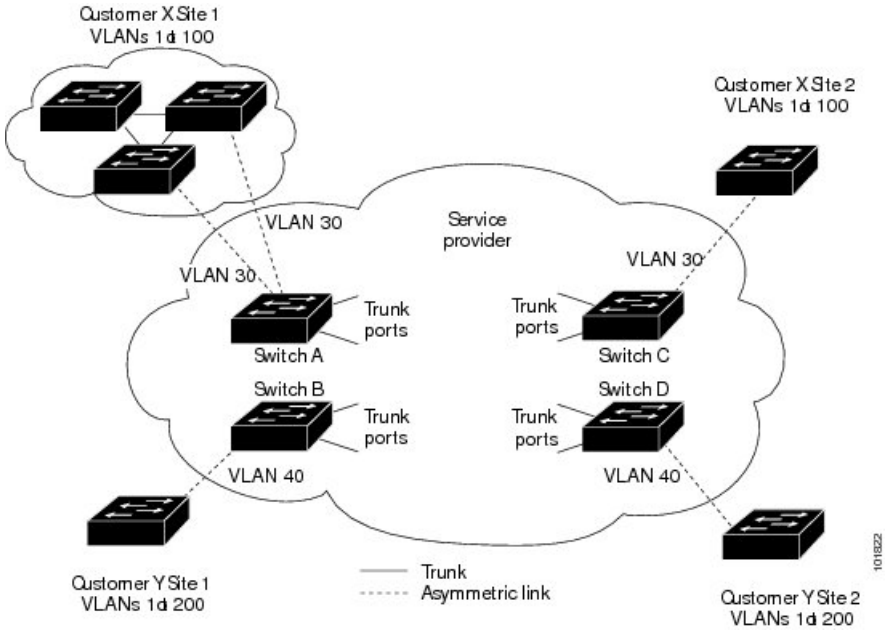
When protocol tunneling is enabled, edge switches on the inbound side of the service-provider infrastructure encapsulate Layer 2 protocol packets with a special MAC address and send them across the service-provider network. Core switches in the network do not process these packets, but forward them as 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 come into the supervisor will cause the CPU load to go up. You might need to make use of software rate limiters to reduce the load on the supervisor CPU. See Configuring Thresholds for Layer 2 Protocol Tunnel Ports, on page 335.

For example, in the figure below, Customer X has four switches in the same VLAN that are connected through the service-provider network. If the network does not tunnel BPDUs, switches on the far ends of the network cannot properly run the STP, CDP, 802.1X, and VTP protocols.

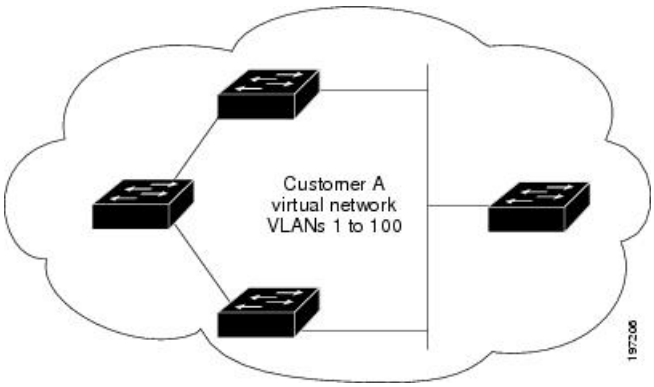
Figure 33: Layer 2 Protocol Tunneling



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 figure below shows the resulting topology on the customer's network when BPDU tunneling is not enabled.

Figure 34: Virtual Network Topology Without BPDU Tunneling



Selective Q-in-Q with Multiple Provider VLANs

Selective Q-in-Q with multiple provider VLANs is a tunneling feature that allows user-specific range of customer VLANs on a port to be associated with one specific provider VLAN and enables you to have multiple customer VLAN to provider VLAN mappings on a port. Packets that come in with a VLAN tag that matches any of the configured customer VLANs on the port are tunneled across the fabric using the properties of the service provider VLAN. The encapsulated packet carries the customer VLAN tag as part of the Layer 2 header of the inner packet.

About Port VLAN Mapping on VLANs (Translating incoming VLANs)

When a service provider has multiple customers connecting to the same physical switch using the same VLAN encapsulation, but they should not be on the same Layer 2 segment, translating the incoming VLAN to a unique VLAN/VNI is the right way to extending the segment.

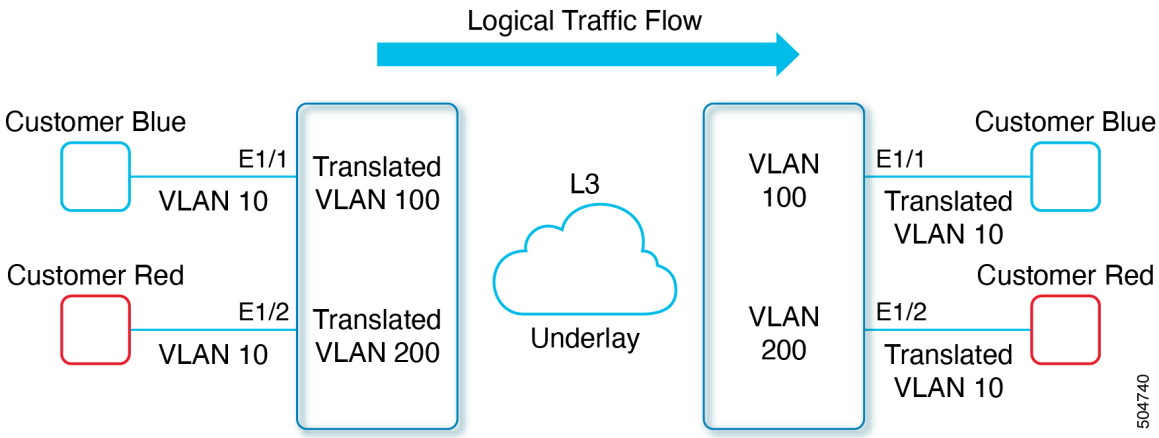
Beginning with Cisco NX-OS Release 10.3(3)F, Port VLAN mapping on non-VXLAN VLANs is supported on Cisco Nexus 9300-EX/FX/FX2/FX3/GX/GX2, C9408 platform switches and Cisco Nexus 9500 switches with 9700-EX/FX/GX line cards.

In the figure below two customers, Blue and Red are connecting to the leaf using VLAN 10 as their encapsulation.

In this example VLAN 10 for Customer Blue (on interface E1/1) is mapped/translated to VLAN 100, and VLAN 10 for customer Red (on interface E1/2) is mapped to VLAN 200.

On the other leaf, this mapping is applied in reverse. Incoming VLAN 100 is mapped to VLAN 10 on Interface E1/1 and VLAN 200 is mapped to VLAN 10 on Interface E1/2.

Figure 35: Logical Traffic Flow



You can configure VLAN translation between the ingress (incoming) VLAN and a local (translated) VLAN on a port. For the traffic arriving on the interface where VLAN translation is enabled, the incoming VLAN is mapped to a translated VLAN.

On the underlay, the inner dot1q is deleted, and switched over to the non-VXLAN network. On the outgoing interface, where VLAN translation is configured, the traffic is converted to the original VLAN and egressed out. Refer to the VLAN counters on the translated VLAN for the traffic counters and not on the ingress VLAN.

Guidelines and Limitations for Q-in-Q tunneling and Layer 2 Protocol Tunneling

Q-in-Q tunnels and Layer 2 tunneling have the following configuration guidelines and limitations:

- Q-in-Q should be configured on the customer-facing interface of the service provider's edge device. If an Ethernet frame ingresses a Cisco Nexus 9000 series switch, the switch cannot encapsulate the frame with two 802.1Q headers within a single forwarding decision. Similarly, if a Q-in-Q-encapsulated Ethernet frame needs to egress a Cisco Nexus 9000 series switch without any 802.1Q headers, the switch cannot decapsulate two 802.1Q headers from the Ethernet frame within a single forwarding decision.
- Mapping multiple VLANs is supported.
- Multiple selective Q-in-Q tags are not supported. That is, Q-in-Q does not support multiple SP tags on a single interface.
- Switches in the service-provider network must be configured to handle the increase in MTU size due 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 might 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.
- The **system dot1q-tunnel transit** command have the following limitations:

  - This commands is required on Cisco Nexus 9300-EX/FX/FX2/FX3/GX switches and 9500 switches with 9700-EX/FX/GX line cards if the device is configured with Q-in-Q, Selective Q-in-Q or Selective Q-in-Q with multiple provider VLAN features.
  - It is required that you configure the **system dot1q-tunnel transit** command on ToR or modular devices.
  - It is required that you configure the **system dot1q-tunnel transit** command on vPC switches or non-vPC switches.
  - Layer 2 frames that exit trunk ports will always be tagged, even with the native VLAN of the port if these commands have been configured.
  - The MPLS, GRE, and IP-in-IP functionalities will not function effectively in conjunction with the Q-in-Q tunneling features if this command is configured on the switch.

- Cisco Nexus 9000 Series devices can provide only MAC-layer ACL/QoS for tunnel traffic (VLAN IDs and src/dest MAC addresses).
- You should use MAC address-based frame distribution.
- Asymmetrical links do not support the Dynamic Trunking Protocol (DTP) because only one port on the link is a trunk. You must configure the 802.1Q trunk port on an asymmetrical link to trunk unconditionally.
- 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.
- You must disable IGMP snooping on the tunnel VLANs.
- You should enter the vlan dot1Q tag native command to maintain the tagging on the native VLAN and drop untagged traffic. This command prevents native VLAN misconfigurations.
- You must manually configure the 802.1Q interfaces to be edge ports.
- IGMP snooping is not supported on the inner VLAN.
- Q-in-Q is not supported on the uplink ports of Cisco Nexus 9332PQ, 9372PX, 9372TX, and 93120TX switches and Cisco Nexus 9396PX, 9396TX, and 93128TX switches with the N9K-M6PQ or N9K-M12PQ generic expansion module (GEM).
- Q-in-Q tunnels might be affected by the limitations of the Application Leaf Engine (ALE) uplink ports on Cisco Nexus 9300 and 9500 Series devices: Limitations for ALE Uplink Ports
- Q-in-Q tagging is not supported.
- Layer 2 protocol tunneling is not supported on Cisco Nexus 9500 Series switches with N9K-X9636C-R, N9K-X9636Q-R, N9K-X9636C-RX line cards.
- Cisco Nexus 9500 Series switches with N9K-X9636C-R, N9K-X9636Q-R, N9K-X9636C-RX line cards, Q-in-Q is supported only on port or port-channel Layer 2 Access VLAN Edge devices.
- FEX configuration is not supported on Q-in-Q ports.
- If the command **l2potocol tunnel stp** is configured on a tunnel interface, the VLAN that you configure on the service provider must be different from that of the customer network.

Guidelines and Limitations for Selective Q-in-Q with Multiple Provider VLANs

- For selective Q-in-Q with multiple provider VLANs, all the existing limitations and guidelines for selective Q-in-Q apply.
- Beginning with Cisco NX-OS Release 9.3(5), selective Q-in-Q with multiple provider VLANs feature is supported on Cisco Nexus N9K-C9316D-GX, N9K-C93600CD-GX, N9K-C9364C-GX switches.
- Selective Q-in-Q with multiple provider VLANs feature is supported on Nexus 9300-EX, 9300-FX, 9300-FX2, 9300-FX3 switches.
- When you enable multiple provider VLANs on a vPC port channel, you must make sure that the configuration is consistent across the vPC peers.
- We recommended not to allow provider VLANs on a regular trunk.
- Only allow native VLAN and provider VLANs on the allowed vlan list of a Selective QinQ trunk interface.
- Selective QinQ trunk VLANs *cannot* be mixed with regular VLANs on the same Selective QinQ trunk interface.
- Port to VLAN mappings (for example: switchport vlan mapping 10 20) is not supported on a port that is configured for selective Q-in-Q with multiple provider VLANs.
- Private VLAN is not supported on a port that is configured for selective Q-in-Q with multiple provider VLANs.
- Only Layer 2 switching is supported.
- Routing on provider VLANs is not supported.
- FEX is not supported for selective Q-in-Q with multiple provider VLANs.
- Selective Q-in-Q with multiple provider VLANs commands not DME-ized.
- When VLAN1 is configured as native VLAN with selective Q-in-Q and selective Q-in-Q with multiple provider tag, traffic on the native VLAN gets dropped. Do not configure VLAN1 as native VLAN when the port is configured with the selective Q-in-Q. When VLAN1 is configured as customer VLAN, then the traffic on VLAN1 gets dropped.

Guidelines and Limitations for Combined Access Port Feature set

- Beginning Cisco NX-OS Release 9.3(3), Combined Access Port Feature set is supported on Cisco Nexus C9348GC-FXP switches with IPv4 underlay.
- The Combined Access Port Feature set consists of the following features:
  - Private VLAN (with secondary isolated)
  - Selective Q-in-Q
  - Port-Security

- All the guidelines and limitations for PVLAN and selective Q-in-Q are applicable for Combined Access Port Feature set also.
- Port mode **private-vlan trunk secondary**is supported on Combined Access Port Feature set.
- When you enable Combined Access Port Feature set on a vPC port channel, you must ensure that the configuration is consistent across the vPC peers.
- We recommend that you enter **system dot1q-tunnel transit** when running the Combined Access Port Feature set.
- Port VLAN mapping (for example: **switchport vlan mapping 10 20**) is not supported.
- Only layer 2 switching is supported on Selective Q-in-Q.
- We dont allow **spanning-tree bpdufilter** to be disabled on the interface when dot1q-tunnel is configured on the interface.
- Only routing is supported on native VLAN of the Combined Access Port Feature

Guidelines and Limitations for Port VLAN Mapping on VLANs

The following are the guidelines and Limitations for Port VLAN Mapping:

- Beginning with Cisco NX-OS Release 10.3(3)F, Port VLAN mapping on VLANs is supported on Cisco Nexus 9300-EX/FX/FX2/FX3/GX/GX2, C9408 platform switches and Cisco Nexus 9500 switches with 9700-EX/FX/GX line cards.
- The ingress (incoming) VLAN does not need to be configured on the switch as a VLAN. The translated VLAN must be configured.
- All Layer 2 source address learning and Layer 2 MAC destination lookup occurs on the translated VLAN. See the VLAN counters on the translated VLAN and not on the ingress (incoming) VLAN.
- Port VLAN mapping routing supports configuring an SVI on the translated VLAN.
- The following example shows incoming VLAN 10 being mapped to local VLAN 100:

```
interface ethernet1/1
switchport vlan mapping 10 100
```

- The following is an example of overlapping VLAN for PV translation. In the first statement, VLAN-102 is a translated VLAN. In the second statement, VLAN-102 the VLAN where it is translated to VLAN-103:

```
interface ethernet1/1
switchport vlan mapping 101 102
switchport vlan mapping 102 103
```

- When adding a member to an existing port channel using the force command, the "mapping enable" configuration must be consistent. For example:

```
Int po 101
switchport vlan mapping enable
switchport vlan mapping 101 10
switchport trunk allowed vlan 10

int eth 1/8
/***No configuration***/
```

Note The switchport VLAN mapping enable command is supported only when the port mode is trunk.

- VLAN mapping helps with VLAN localization to a port, scoping the VLANs per port. A typical use case is in the service provider environment where the service provider leaf switch has different customers with overlapping VLANs that come in on different ports. For example, customer A has VLAN 10 coming in on Eth 1/1 and customer B has VLAN 10 coming in on Eth 2/2.
- Port VLAN mapping does not coexist with PVLAN.
- If the **inherit port-profile** command is configured on a PV interface, use the **no inherit port-profile** <*profile name*> command to detach and then execute the **no switchport vlan mapping all** command.
- If the **system dot1q-tunnel transit vlan provider_vlan_list** command is globally configured on the switch, do not set the provider VLAN as the native or access port VLAN for any other trunk or access port on the system. It is expected to choose provider VLANs other than the native VLANs on the system.

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 provider VLAN membership of the port is changed using 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.

For seamless packet forwarding and preservation of all VLAN tags on pure transit boxes in the SP cloud that have no Q-in-Q encapsulation or decapsulation requirement, configure the system-wide **system dot1q-tunnel transit** command. To remove the configuration, use the **no system dot1q-tunnel transit** command.

For the supported platforms and limitations of the **system dot1q-tunnel transit** or **system dot1q-tunnel transit vlan** *provider_vlan_list* command, see Guidelines and Limitations for Q-in-Q tunneling and Layer 2 Protocol Tunneling , on page 325 section.

Before you begin

You must first configure the interface as a switchport.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface ethernet** *slot/port*
3. switch(config-if)# **switchport**

4. switch(config-if)# **switchport mode dot1q-tunnel**
5. switch(config-if)# **spanning-tree port type edge**
6. switch(config-if)# **switchport access vlan** *vlan-id*
7. (Optional) switch(config-if)# **no switchport mode dot1q-tunnel**
8. switch(config-if)# **exit**
9. (Optional) switch(config)# **show dot1q-tunnel** [**interface** *if-range*]
10. (Optional) switch(config)# **no shutdown**
11. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

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 a 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)# **spanning-tree port type edge**	Designates the port as a spanning-tree edge port.
Step 6	switch(config-if)# **switchport access vlan** *vlan-id*	Configures the Provider access VLAN value.
Step 7	(Optional) switch(config-if)# **no switchport mode dot1q-tunnel**	Disables the 802.1Q tunnel on the port.
Step 8	switch(config-if)# **exit**	Exits configuration mode.
Step 9	(Optional) switch(config)# **show dot1q-tunnel** [**interface** *if-range*]	Displays all ports that are in dot1q-tunnel mode. Optionally, you can specify an interface or range of interfaces to display.
Step 10	(Optional) switch(config)# **no shutdown**	Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.
Step 11	(Optional) switch(config)# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

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)# spanning-tree port type edge
switch(config-if)# switchport access vlan vlan 10
switch(config-if)# exit
switch(config)# exit
switch# show dot1q-tunnel
```

Configuring Selective Q-in-Q with Multiple provider VLANs

Before you begin

You must configure provider VLANs

You must disable spanning-tree on the trunk port using the **spanning-tree bpdufilter enable** command.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface** *interface-id*
3. switch(config if)# **switchport**
4. switch(config-if)# **switchport mode trunk**
5. switch(config-if)# **spanning-tree bpdufilter enable**
6. switch(config-if)# **switchport trunk native vlan** *vlan-id*
7. switch(config-if)# **switchport vlan mapping** *vlan-id-range* **dot1q-tunnel** *outer vlan-id*
8. switch(config-if)# **switchport trunk allowed vlan** *vlan_list*
9. switch(config-if)# **exit**
10. switch(config-if)# **show interfaces** *interface-id* **vlan mapping**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface** *interface-id*	Enters interface configuration mode for the interface connected to the service provider network. You can enter a physical interface or an EtherChannel port channel.
Step 3	switch(config if)# **switchport**	Sets the interface as a Layer 2 switching port.
Step 4	switch(config-if)# **switchport mode trunk**	Sets the interface as a Layer 2 trunk port.

	Command or Action	Purpose
Step 5	switch(config-if)# **spanning-tree bpdufilter enable**	Disables the sending and processing of spanning-tree BPDUs on this interface.
Step 6	switch(config-if)# **switchport trunk native vlan** *vlan-id*	Sets the native VLAN for the 802.1Q trunk. Valid values are from 1 to 4094. The default value is VLAN1.
Step 7	switch(config-if)# **switchport vlan mapping** *vlan-id-range* **dot1q-tunnel** *outer vlan-id*	Enter the VLAN IDs to be mapped: • vlan-id-range—The customer VLAN ID range(C-VLAN) entering the switch from the customer network. The range is from 1 to 4094. You can enter a string of VLAN-IDs. • outer vlan-id—Enter the outer VLAN ID (S-VLAN) of the service provider network. The range is from 1 to 4094.
Step 8	switch(config-if)# **switchport trunk allowed vlan** *vlan_list*	Sets the allowed VLANs for the trunk interface.
Step 9	switch(config-if)# **exit**	Exits the configuration mode.
Step 10	switch(config-if)# **show interfaces** *interface-id* **vlan mapping**	Verifies the mapping configuration.

The following example shows how to configure selective Q-in-Q with multiple provider VLANs:

Example

```
switch# sh run int e1/1

interface Ethernet1/1
  switchport
  switchport mode trunk
  switchport trunk native vlan 2
  switchport vlan mapping 3-400 dot1q-tunnel 400
  switchport vlan mapping 401-800 dot1q-tunnel 401
  switchport vlan mapping 801-1200 dot1q-tunnel 10
  switchport vlan mapping 1201-1600 dot1q-tunnel 1400
  switchport vlan mapping 1601-2000 dot1q-tunnel 9
  switchport vlan mapping 2001-2400 dot1q-tunnel 3000
  switchport vlan mapping 2401-2800 dot1q-tunnel 2099
  switchport vlan mapping 2801-3200 dot1q-tunnel 2800
  switchport vlan mapping 3201-3600 dot1q-tunnel 3967
  switchport vlan mapping 3601-4000 dot1q-tunnel 600
  spanning-tree bpdufilter enable
  switchport trunk allowed vlan 2,9-10,400-401,600,1400,2099,2800,3000,3967

switch# show interface e1/1 vlan mapping
Interface Eth1/1:
Original VLAN                     Translated VLAN
---------------                   -----------------
3                                        400
4                                        400
5                                        400
6                                        400
```

```
7                                               400
8                                               400
9                                               400
10                                              400
11                                              400
12                                              400
13                                              400
14                                              400
15                                              400
16                                              400
17                                              400
18                                              400
19                                              400
20                                              400

switch# show consistency-checker selective-qinq interface e1/1
Fetching ingressVlanXlate entries from slice:0 HW
Fetching ingressVlanXlate entries from slice:1 HW
Performing port specific checks for intf Eth1/1
Port specific selective QinQ checks for interface   Eth1/1 : PASS

Switch#
```

Changing the EtherType for Q-in-Q

The switch default EtherType is 0x8100 for 802.1Q and Q-in-Q encapsulations. EtherType cannot be configured to 0x9100, 0x9200 and 0x88a8 on the switchport interface.

Enabling the Layer 2 Protocol Tunnel

You can enable protocol tunneling on the 802.1Q tunnel port.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface ethernet** *slot/port*
3. switch(config-if)# **switchport**
4. switch(config-if)# **switchport mode dot1q-tunnel**
5. switch(config-if)# **l2protocol tunnel** [**cdp** | **stp** | **lacp** | **lldp** |**vtp**]
6. (Optional) switch(config-if)# **no l2protocol tunnel** [**cdp** | **stp** | **lacp** | **lldp** |**vtp**]
7. switch(config-if)# **exit**
8. (Optional) switch(config)# **no shutdown**
9. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

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*	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 a 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)# **l2protocol tunnel** [**cdp** | **stp** | **lacp** | **lldp** |**vtp**]	Enables Layer 2 protocol tunneling. Optionally, you can enable CDP, STP, LACP, LLDP, or VTP tunneling.
Step 6	(Optional) switch(config-if)# **no l2protocol tunnel** [**cdp** | **stp** | **lacp** | **lldp** |**vtp**]	Disables protocol tunneling.
Step 7	switch(config-if)# **exit**	Exits configuration mode.
Step 8	(Optional) switch(config)# **no shutdown**	Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.
Step 9	(Optional) switch(config)# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

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 Global CoS for L2 Protocol Tunnel Ports

You can specify a Class of Service (CoS) value globally so that ingress BPDUs on the tunnel ports are encapsulated with the specified class.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **l2protocol tunnel cos** *value*
3. (Optional) switch(config)# **no l2protocol tunnel cos**
4. switch(config)# **exit**

5. (Optional) switch# **no shutdown**
6. (Optional) switch# **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **l2protocol tunnel cos** *value*	Specifies a global CoS value on all Layer 2 protocol tunneling ports. The default cos-value is 5.
Step 3	(Optional) switch(config)# **no l2protocol tunnel cos**	Sets the global CoS value to default.
Step 4	switch(config)# **exit**	Exits configuration mode.
Step 5	(Optional) switch# **no shutdown**	Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.
Step 6	(Optional) switch# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to specify a global CoS value for the purpose of Layer 2 protocol tunneling:

```
switch# configure terminal
switch(config)# l2protocol tunnel cos 6
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.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface ethernet** *slot/port*
3. switch(config-if)# **switchport**
4. switch(config-if)# **switchport mode dot1q-tunnel**
5. switch(config-if)# **l2protocol tunnel drop-threshold** [**cdp** | **stp** | **vtp**] *packets-per-sec*
6. (Optional) switch(config-if)# **no l2protocol tunnel drop-threshold** [**cdp** | **stp** | **vtp**]
7. switch(config-if)# **l2protocol tunnel shutdown-threshold** [**cdp** | **stp** | **vtp**] *packets-per-sec*
8. (Optional) switch(config-if)# **no l2protocol tunnel shutdown-threshold** [**cdp** | **stp** | **vtp**]

9. switch(config-if)# **exit**
10. (Optional) switch(config)# **no shutdown**
11. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

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 a 802.1Q tunnel on the port.
Step 5	switch(config-if)# **l2protocol tunnel drop-threshold** [**cdp** | **stp** | **vtp**] *packets-per-sec*	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.
Step 6	(Optional) switch(config-if)# **no l2protocol tunnel drop-threshold** [**cdp** | **stp** | **vtp**]	Resets the threshold values to 0 and disables the drop threshold.
Step 7	switch(config-if)# **l2protocol tunnel shutdown-threshold** [**cdp** | **stp** | **vtp**] *packets-per-sec*	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 CDP, STP, or VTP. Valid values for the packets is from 1 to 4096.
Step 8	(Optional) switch(config-if)# **no l2protocol tunnel shutdown-threshold** [**cdp** | **stp** | **vtp**]	Resets the threshold values to 0 and disables the shutdown threshold.
Step 9	switch(config-if)# **exit**	Exits configuration mode.
Step 10	(Optional) switch(config)# **no shutdown**	Clears the errors on the interfaces and VLANs where policies correspond with hardware policies. This command allows policy programming to continue and the port to come up. If policies do not correspond, the errors are placed in an error-disabled policy state.
Step 11	(Optional) switch(config)# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Configuring Combined Access Port Feature set

To configure combined access port feature set follow these steps.

SUMMARY STEPS

1. **interface** *interface* [**port** | **port-channel** | **vPC**]
2. **switchport mode private-vlan trunk** *secondary*
3. **switchport private-vlan trunk native vlan** *vlan_id*
4. **switchport private-vlan trunk allowed vlan** *vlan list*
5. **switchport private-vlan association trunk** *primary_vlan_ID secondary_vlan_ID*
6. **switchport vlan mapping** [*vlan-id-range* | *all*] *dot1q-tunnel outer vlan-id*
7. **storm-control broadcast level** [*high level*] [*lower level*]
8. **storm-control multicast level** [*high level*] [*lower level*]
9. **storm-control action** [**shutdown** | **trap**]
10. **load-interval counter** {*1* | *2* | *3* }
11. **switchport port-security maximum** [**max-addr**]
12. **switchport port-security** *action* [**restrict** | **shutdown** | **protect**]
13. **switchport port-security**
14. **service-policy** {**input** | **type** {**qos input** | **queuing** {**input** | **output**}}} *policy-map-name*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**interface** *interface* [**port**	**port-channel**	**vPC**] **Example:** `switch# interface port-channel 202`	Places you into the interface configuration mode for the specified port channel. The range is from 1 to 4096.
Step 2	**switchport mode private-vlan trunk** *secondary* **Example:** `switch(config)# switchport mode private-vlan trunk secondary`	Configures the port as a secondary trunk port for a private VLAN.
Step 3	**switchport private-vlan trunk native vlan** *vlan_id* **Example:** `switch(config)# switchport private-vlan trunk native vlan 4002`	Configures native VLAN assigned on a PVLAN trunk port.
Step 4	**switchport private-vlan trunk allowed vlan** *vlan list* **Example:** `switch(config)# switchport private-vlan trunk allowed vlan 1002,4002`	Configures a list of allowed normal VLANs on a PVLAN trunk port.
Step 5	**switchport private-vlan association trunk** *primary_vlan_ID secondary_vlan_ID* **Example:** `switch(config)# switchport private-vlan association trunk 4050 4049`	Configures association between primary VLAN and secondary VLAN on the PVLAN trunk port.

	Command or Action	Purpose
Step 6	**switchport vlan mapping** [*vlan-id-range* | *all*] *dot1q-tunnel outer vlan-id* **Example:** `switch(config-if)# switchport vlan mapping all dot1q-tunnel 1002`	Enter the customer range VLANs or keyword all which includes all the 4K VLANs.
Step 7	**storm-control broadcast level** [*high level*] [*lower level*] **Example:** `switch(config-if)# storm-control broadcast level 1.00`	Configures broadcast storm control. Specifies the upper threshold levels for broadcast traffic.
Step 8	**storm-control multicast level** [*high level*] [*lower level*] **Example:** `switch(config-if)# storm-control multicast level 1.00`	Enables multicast traffic storm control on the interface, configures the traffic storm control level, and applies the traffic storm control level to all traffic storm control modes enabled on the interface.
Step 9	**storm-control action** [**shutdown** | **trap**] **Example:** `switch(config-if)# storm-control action shutdown`	Configures traffic storm-control to either generate trap or error-disable the port when a traffic storm occurs.
Step 10	**load-interval counter** {*1* | *2* | *3* } **Example:** `switch(config-if)# load-interval counter 1 5`	Specifies the interval between sampling statistics on the interface.
Step 11	**switchport port-security maximum** [**max-addr**] **Example:** `switch(config-if)# switchport port-security maximum 3`	Sets the maximum number of secure MAC addresses on a port.
Step 12	**switchport port-security** *action* [**restrict** | **shutdown** | **protect**] **Example:** `switch(config-if)# switchport port-security violation restrict`	Restrict security violation mode on the interface.
Step 13	**switchport port-security** **Example:** `switch(config-if)# switchport port-security`	Displays the port security configuration information.
Step 14	**service-policy** {**input** | **type** {**qos input** | **queuing** {**input** | **output**}}} *policy-map-name* **Example:** `switch(config-if)# service-policy type qos input ovh_qos`	Attaches a policy map to an interface.

Configuring Q-in-Q Double Tagging

Enable multi-tagging for STP and CDP BPDUs.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface*
3. **switchport**
4. **switchport mode dot1q-tunnel**
5. **l2protocol tunnel [cdp | stp]**
6. (Optional) **no l2protocol tunnel [cdp | stp]**
7. **l2protocol tunnel allow-double-tag**
8. (Optional) **no l2protocol tunnel allow-double-tag**
9. **exit**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**interface** *interface* **Example:** `switch(config)# interface ethernet 7/1`	Specifies the interface that you are configuring.
Step 3	**switchport** **Example:** `switch(config-if)# switchport`	Sets the interface as a Layer 2 switching port.
Step 4	**switchport mode dot1q-tunnel** **Example:** `switch(config-if)# switchport mode dot1q-tunnel`	Creates an 802.1Q tunnel on the port. The port goes down and reinitializes (port flap) when the interface mode is changed. BPDU filtering is enabled and CDP is disabled on tunnel interfaces.
Step 5	**l2protocol tunnel [cdp | stp]** **Example:** `switch(config-if)# l2protocol tunnel cdp`	Enables Layer 2 protocol tunneling. Optionally, you can enable CDP or STP.
Step 6	(Optional) **no l2protocol tunnel [cdp | stp]** **Example:** `switch(config-if)# no l2protocol tunnel stp`	Disables protocol tunneling.

	Command or Action	Purpose
Step 7	**l2protocol tunnel allow-double-tag** **Example:** `switch(config-if)# l2protocol tunnel allow-double-tag`	Enables multi-tagging for STP and CDP BPDUs on the interface.
Step 8	(Optional) **no l2protocol tunnel allow-double-tag** **Example:** `switch(config-if)# no l2protocol tunnel allow-double-tag`	Disables multi-tagging for STP and CDP BPDUs on the interface.
Step 9	**exit** **Example:** `switch(config-if)# exit`	Exits configuration mode.

Example

This example shows how to enable multi-tagging for STP and CDP BPDUs:

```
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 cdp
switch(config-if)#  l2protocol tunnel stp
switch(config-if)#  l2protocol tunnel allow-double-tag
switch(config-if)# exit
switch(config)# exit
switch#
```

Verifying the Q-in-Q Configuration

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, for 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.

Command	Purpose
show running-config l2pt	Displays the current Layer 2 protocol tunnel running configuration.

Configuration Examples 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)# no ip igmp snooping
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#
```

Configuring Port VLAN Mapping on VLANs

Before you begin

- Ensure that the physical or port channel on which you want to implement VLAN translation is configured as a Layer 2 trunk port.
- Ensure that the translated VLANs are created on the switch and are also added to the Layer 2 trunk ports trunk-allowed VLAN vlan-list.

Note As a best practice, do not add the ingress VLAN ID to the switchport allowed vlan-list under the interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type/port*

3. **[no] switchport vlan mapping enable**
4. **[no] switchport vlan mapping** *vlan-id translated-vlan-id*
5. **[no] switchport vlan mapping all**
6. **copy running-config startup-config**
7. **show interface** [*if-identifier*] **vlan mapping**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**interface** *type/port* **Example:** `switch(config)# interface Ethernet1/1`	Specifies the interface that you are configuring.
Step 3	**[no] switchport vlan mapping enable** **Example:** `switch(config-if)# [no] switchport vlan mapping enable`	Enables VLAN translation on the switch port. VLAN translation is disabled by default. **Note** Use the **no** form of this command to disable VLAN translation.
Step 4	**[no] switchport vlan mapping** *vlan-id translated-vlan-id* **Example:** `switch(config-if)# switchport vlan mapping 10 100`	Translates a VLAN to another VLAN. • The range for both the *vlan-id* and *translated-vlan-id* arguments are from 1 to 4094. • You can configure VLAN translation between the ingress (incoming) VLAN and a local (translated) VLAN on a port. For the traffic arriving on the interface where VLAN translation is enabled, the incoming VLAN is mapped to a translated VLAN. Routing of traffic happens in context of SVI for translated VLAN. On the outgoing interface, where VLAN translation is configured, the traffic is converted to the original VLAN and egresses out. **Note** Use the **no** form of this command to clear the mappings between a pair of VLANs.
Step 5	**[no] switchport vlan mapping all** **Example:** `switch(config-if)# no switchport vlan mapping all`	Removes all VLAN mappings configured on the interface.

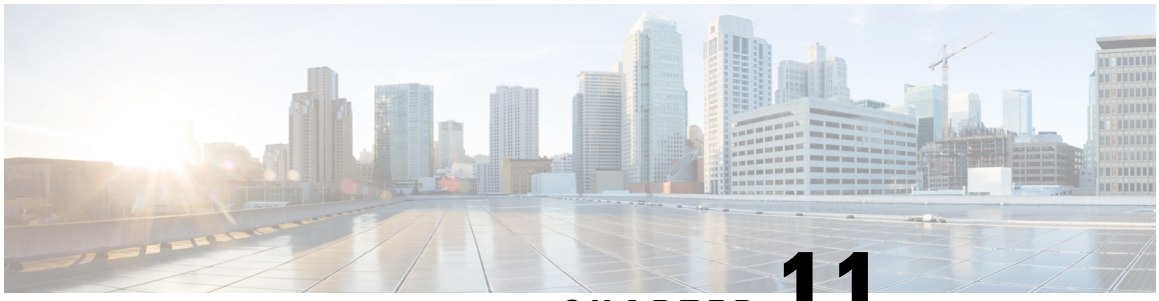
	Command or Action	Purpose
Step 6	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Copies the running configuration to the startup configuration. **Note** The VLAN translation configuration does not become effective until the switch port becomes an operational trunk port.
Step 7	**show interface** [*if-identifier*] **vlan mapping** **Example:** `switch# show interface ethernet1/1 vlan mapping`	Displays VLAN mapping information for a range of interfaces or for a specific interface.

Example

This example shows how to configure VLAN translation between (the ingress) VLAN 10 and (the local) VLAN 100. The show vlan counters command output shows the statistic counters as translated VLAN instead of customer VLAN.

```
switch#  configure terminal
switch(config)#  interface ethernet1/1
switch(config-if)#  switchport vlan mapping enable
switch(config-if)#  switchport vlan mapping 10 100
switch(config-if)#  switchport trunk allowed vlan 100
switch(config-if)#  show interface ethernet1/1 vlan mapping
Interface eth1/1:
Original VLAN           Translated VLAN
------------------      ---------------
10                           100

switch(config-if)#  show vlan counters
Vlan Id                              :100
Unicast Octets In                    :292442462
Unicast Packets In                   :1950525
Multicast Octets In                  :14619624
Multicast Packets In                 :91088
Broadcast Octets In                  :14619624
Broadcast Packets In                 :91088
Unicast Octets Out                   :304012656
Unicast Packets Out                  :2061976
L3 Unicast Octets In                 :0
L3 Unicast Packets In                :0
```



CHAPTER 11

Configuring Port VLAN Mapping on VLANs

This chapter contains these sections:

About Port VLAN Mapping on VLANs (Translating incoming VLANs)

When a service provider has multiple customers connecting to the same physical switch using the same VLAN encapsulation, but they should not be on the same Layer 2 segment, translating the incoming VLAN to a unique VLAN/VNI is the right way to extending the segment.

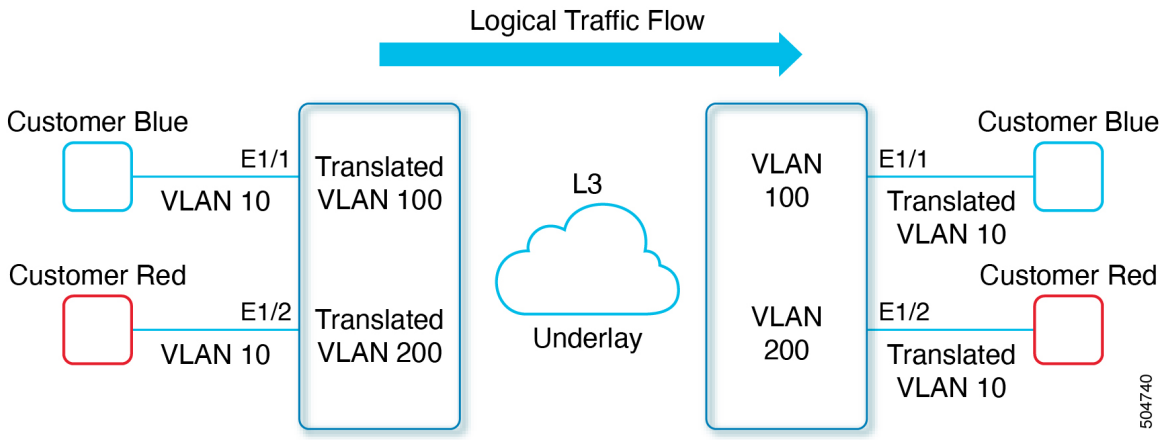
Beginning with Cisco NX-OS Release 10.3(3)F, Port VLAN mapping on non-VXLAN VLANs is supported on Cisco Nexus 9300-EX/FX/FX2/FX3/GX/GX2, C9408 platform switches and Cisco Nexus 9500 switches with 9700-EX/FX/GX line cards.

In the figure below two customers, Blue and Red are connecting to the leaf using VLAN 10 as their encapsulation.

In this example VLAN 10 for Customer Blue (on interface E1/1) is mapped/translated to VLAN 100, and VLAN 10 for customer Red (on interface E1/2) is mapped to VLAN 200.

On the other leaf, this mapping is applied in reverse. Incoming VLAN 100 is mapped to VLAN 10 on Interface E1/1 and VLAN 200 is mapped to VLAN 10 on Interface E1/2.

Figure 36: Logical Traffic Flow



You can configure VLAN translation between the ingress (incoming) VLAN and a local (translated) VLAN on a port. For the traffic arriving on the interface where VLAN translation is enabled, the incoming VLAN is mapped to a translated VLAN.

On the underlay, the inner dot1q is deleted, and switched over to the non-VXLAN network. On the outgoing interface, where VLAN translation is configured, the traffic is converted to the original VLAN and egressed out. Refer to the VLAN counters on the translated VLAN for the traffic counters and not on the ingress VLAN.

Guidelines and Limitations for Port VLAN Mapping on VLANs

The following are the guidelines and Limitations for Port VLAN Mapping:

- Beginning with Cisco NX-OS Release 10.3(3)F, Port VLAN mapping on VLANs is supported on Cisco Nexus 9300-EX/FX/FX2/FX3/GX/GX2, C9408 platform switches and Cisco Nexus 9500 switches with 9700-EX/FX/GX line cards.
- The ingress (incoming) VLAN does not need to be configured on the switch as a VLAN. The translated VLAN must be configured.
- All Layer 2 source address learning and Layer 2 MAC destination lookup occurs on the translated VLAN. See the VLAN counters on the translated VLAN and not on the ingress (incoming) VLAN.
- Port VLAN mapping routing supports configuring an SVI on the translated VLAN.
- The following example shows incoming VLAN 10 being mapped to local VLAN 100:

```
interface ethernet1/1
switchport vlan mapping 10 100
```

- The following is an example of overlapping VLAN for PV translation. In the first statement, VLAN-102 is a translated VLAN. In the second statement, VLAN-102 the VLAN where it is translated to VLAN-103:

```
interface ethernet1/1
switchport vlan mapping 101 102
switchport vlan mapping 102 103
```

- When adding a member to an existing port channel using the force command, the "mapping enable" configuration must be consistent. For example:

```
Int po 101
switchport vlan mapping enable
switchport vlan mapping 101 10
switchport trunk allowed vlan 10

int eth 1/8
/***No configuration***/
```

Note The switchport VLAN mapping enable command is supported only when the port mode is trunk.

- VLAN mapping helps with VLAN localization to a port, scoping the VLANs per port. A typical use case is in the service provider environment where the service provider leaf switch has different customers with overlapping VLANs that come in on different ports. For example, customer A has VLAN 10 coming in on Eth 1/1 and customer B has VLAN 10 coming in on Eth 2/2.
- Port VLAN mapping does not coexist with PVLAN.
- If the **inherit port-profile** command is configured on a PV interface, use the **no inherit port-profile** <*profile name*> command to detach and then execute the **no switchport vlan mapping all** command.
- If the **system dot1q-tunnel transit vlan provider_vlan_list** command is globally configured on the switch, do not set the provider VLAN as the native or access port VLAN for any other trunk or access port on the system. It is expected to choose provider VLANs other than the native VLANs on the system.

Configuring Port VLAN Mapping on VLANs

Before you begin

- Ensure that the physical or port channel on which you want to implement VLAN translation is configured as a Layer 2 trunk port.
- Ensure that the translated VLANs are created on the switch and are also added to the Layer 2 trunk ports trunk-allowed VLAN vlan-list.

Note As a best practice, do not add the ingress VLAN ID to the switchport allowed vlan-list under the interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type/port*
3. **[no] switchport vlan mapping enable**
4. **[no] switchport vlan mapping** *vlan-id translated-vlan-id*
5. **[no] switchport vlan mapping all**
6. **copy running-config startup-config**

7. **show interface** [*if-identifier*] **vlan mapping**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**interface** *type/port* **Example:** `switch(config)# interface Ethernet1/1`	Specifies the interface that you are configuring.
Step 3	**[no] switchport vlan mapping enable** **Example:** `switch(config-if)# [no] switchport vlan mapping enable`	Enables VLAN translation on the switch port. VLAN translation is disabled by default. **Note** Use the **no** form of this command to disable VLAN translation.
Step 4	**[no] switchport vlan mapping** *vlan-id translated-vlan-id* **Example:** `switch(config-if)# switchport vlan mapping 10 100`	Translates a VLAN to another VLAN. • The range for both the *vlan-id* and *translated-vlan-id* arguments are from 1 to 4094. • You can configure VLAN translation between the ingress (incoming) VLAN and a local (translated) VLAN on a port. For the traffic arriving on the interface where VLAN translation is enabled, the incoming VLAN is mapped to a translated VLAN. Routing of traffic happens in context of SVI for translated VLAN. On the outgoing interface, where VLAN translation is configured, the traffic is converted to the original VLAN and egresses out. **Note** Use the **no** form of this command to clear the mappings between a pair of VLANs.
Step 5	**[no] switchport vlan mapping all** **Example:** `switch(config-if)# no switchport vlan mapping all`	Removes all VLAN mappings configured on the interface.
Step 6	**copy running-config startup-config** **Example:**	Copies the running configuration to the startup configuration. **Note**

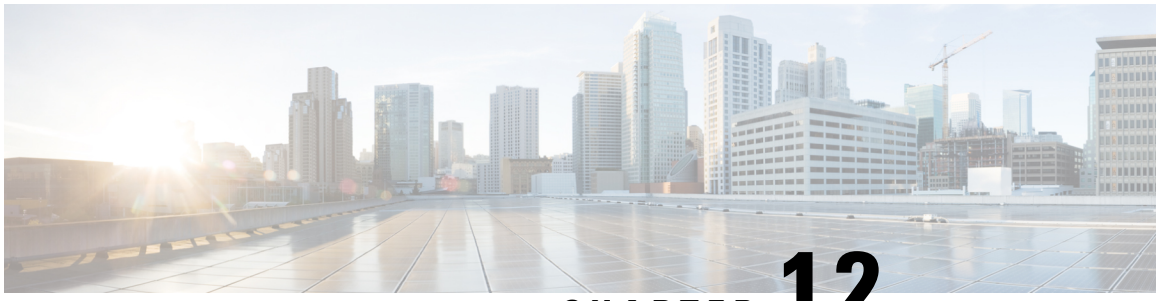
	Command or Action	Purpose
	`switch(config-if)# copy running-config startup-config`	The VLAN translation configuration does not become effective until the switch port becomes an operational trunk port.
Step 7	**show interface** [*if-identifier*] **vlan mapping** **Example:** `switch# show interface ethernet1/1 vlan mapping`	Displays VLAN mapping information for a range of interfaces or for a specific interface.

Example

This example shows how to configure VLAN translation between (the ingress) VLAN 10 and (the local) VLAN 100. The show vlan counters command output shows the statistic counters as translated VLAN instead of customer VLAN.

```
switch#  configure terminal
switch(config)#  interface ethernet1/1
switch(config-if)#  switchport vlan mapping enable
switch(config-if)#  switchport vlan mapping 10 100
switch(config-if)#  switchport trunk allowed vlan 100
switch(config-if)#  show interface ethernet1/1 vlan mapping
Interface eth1/1:
Original VLAN           Translated VLAN
------------------      ---------------
10                           100

switch(config-if)#  show vlan counters
Vlan Id                              :100
Unicast Octets In                    :292442462
Unicast Packets In                   :1950525
Multicast Octets In                  :14619624
Multicast Packets In                 :91088
Broadcast Octets In                  :14619624
Broadcast Packets In                 :91088
Unicast Octets Out                   :304012656
Unicast Packets Out                  :2061976
L3 Unicast Octets In                 :0
L3 Unicast Packets In                :0
```



CHAPTER 12

Configuring Static and Dynamic NAT Translation

Network Address Translation Overview

Network Address Translation (NAT) enables private IP internetworks that use nonregistered IP addresses to connect to the Internet. NAT operates on a device, usually connecting two networks, and translates private (not globally unique) IP addresses in the internal network into legal IP addresses before packets are forwarded to another network. You can configure NAT to advertise only one IP address for the entire network to the outside world. This ability provides additional security, effectively hiding the entire internal network behind one IP address.

A device configured with NAT has at least one interface to the inside network and one to the outside network. In a typical environment, NAT is configured at the exit router between a stub domain and a backbone. When a packet leaves the domain, NAT translates the locally significant source IP address into a globally unique IP address. When a packet enters the domain, NAT translates the globally unique destination IP address into a local IP address. If more than one exit point exists, NAT configured at each point must have the same translation table.

NAT is described in RFC 1631.

Information About Static NAT

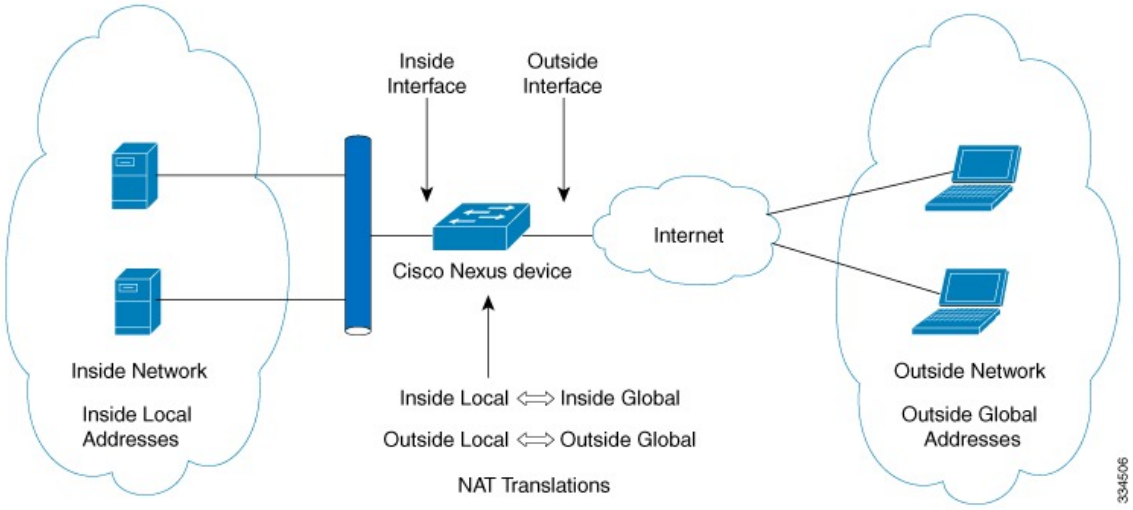
Static Network Address Translation (NAT) allows the user to configure one-to-one translations of the inside local addresses to the outside global addresses. It allows both IP addresses and port number translations from the inside to the outside traffic and the outside to the inside traffic. The Cisco Nexus device supports Hitless NAT, which means that you can add or remove a NAT translation in the NAT configuration without affecting the existing NAT traffic flows.

Static NAT creates a fixed translation of private addresses to public addresses. Because static NAT assigns addresses on a one-to-one basis, you need an equal number of public addresses as private addresses. Because the public address is the same for each consecutive connection with static NAT, and a persistent translation rule exists, static NAT enables hosts on the destination network to initiate traffic to a translated host if an access list exists that allows it .

With dynamic NAT and Port Address Translation (PAT), each host uses a different address or port for each subsequent translation. The main difference between dynamic NAT and static NAT is that static NAT allows a remote host to initiate a connection to a translated host if an access list exists that allows it, while dynamic NAT does not.

The figure shows a typical static NAT scenario. The translation is always active so both translated and remote hosts can originate connections, and the mapped address is statically assigned by the **static** command.

Figure 37: Static NAT



These are key terms to help you understand static NAT:

- NAT inside interface—The Layer 3 interface that faces the private network.
- NAT outside interface—The Layer 3 interface that faces the public network.
- Local address—Any address that appears on the inside (private) portion of the network.
- Global address—Any address that appears on the outside (public) portion of the network.
- Legitimate IP address—An address that is assigned by the Network Information Center (NIC) or service provider.

- Inside local address—The IP address assigned to a host on the inside network. This address does not need to be a legitimate IP address.
- Outside local address—The IP address of an outside host as it appears to the inside network. It does not have to be a legitimate address, because it is allocated from an address space that can be routed on the inside network.
- Inside global address—A legitimate IP address that represents one or more inside local IP addresses to the outside world.
- Outside global address—The IP address that the host owner assigns to a host on the outside network. The address is a legitimate address that is allocated from an address or network space that can be routed.

Dynamic NAT Overview

Dynamic Network Address Translation (NAT) translates a group of real IP addresses into mapped IP addresses that are routable on a destination network. Dynamic NAT establishes a one-to-one mapping between unregistered and registered IP addresses; however, the mapping can vary depending on the registered IP address that is available at the time of communication.

A dynamic NAT configuration automatically creates a firewall between your internal network and outside networks or the Internet. Dynamic NAT allows only connections that originate inside the stub domain—a device on an external network cannot connect to devices in your network, unless your device has initiated the contact.

Dynamic NAT translations do not exist in the NAT translation table until a device receives traffic that requires translation. Dynamic translations are cleared or timed out when not in use to make space for new entries. Usually, NAT translation entries are cleared when the ternary content addressable memory (TCAM) entries are limited. The default minimum timeout for dynamic NAT translations is 30 minutes.

Note The **ip nat translation sampling-timeout** command is not supported. Statistics are collected every 60 seconds for the installed NAT policies. These statistics are used to determine if the flow is active or not.

Dynamic NAT supports Port Address Translation (PAT) and access control lists (ACLs). PAT, also known as overloading, is a form of dynamic NAT that maps multiple unregistered IP addresses to a single registered IP address by using different ports. Your NAT configuration can have multiple dynamic NAT translations with same or different ACLs. However, for a given ACL, only one interface can be specified.

Timeout Mechanisms

Timeout mechanisms are configurable timers that control how long certain NAT translation timeout entries are maintained before being cleared or expired. You need to carve out a separate TCP-NAT TCAM region before configuring the timers.

Note The TCP-NAT tcam region is separate from the standard NAT TCAM region.

The following NAT translation timeout timers are supported on the switch:

- **syn-timeout** - Timeout value for TCP data packets that send the SYN request, but do not receive a SYN-ACK reply.

  The timeout value ranges from 1 second to 172800 seconds. The default value is 60 seconds when the TCP-NAT tcam region is carved. If TCP-NAT TCAM region is *not* carved, the default value is set to never.

Note The **syn-timeout** option is supported only on Cisco Nexus 9200 and 9300-EX, -FX, -FX2, -FX3, -FXP, -GX platform switches.

- **finrst-timeout** - Timeout value for the flow entries when a connection is terminated by receiving RST or FIN packets. Use the same keyword to configure the behavior for both RST and FIN packets.

  The timeout value ranges from 1 second to 172800 seconds. The default value is 60 seconds when the TCP-NAT tcam region is carved. If TCP-NAT TCAM region is *not* carved, the default value is set to never.

  - If a FIN packet is received after the connection is established, SYN-->SYN-ACK-->FIN, the finrst timer starts.
  - If a FIN-ACK is received from the other side, the translation entry is cleared immediately, else it clears after the timeout value completes.

    The timeout value ranges from 1 second to 172800 seconds. The default value is 60 seconds.
  - If an RST packet is received after the connection is established, SYN-->SYN-ACK-->RST, the translation entry is cleared immediately.

Note If dynamic pool-based configuration is used and a FIN-ACK is received, the translation entry is not cleared.

Note The **finrst-timeout** option is supported only on Cisco Nexus 9200 and 9300-EX, -FX, -FX2, -FX3, -FXP, -GX platform switches.

- **tcp-timeout** - Timeout value for TCP translations for which connections have been established after a three-way handshake (SYN, SYN-ACK, ACK). If no active flow occurs after the connection has been established, the translations expire as per the configured timeout value.

  The timeout value ranges from 60 seconds to 172800 seconds. The default value is 3600 seconds.

- **udp-timeout** - Timeout value for all NAT UDP packets.

  The timeout value ranges from 60 seconds to 172800 seconds. The default value is 3600 seconds.

- **timeout** - Timeout value for dynamic NAT translations.

  The timeout value ranges from 60 seconds to 172800 seconds. The default value is 3600 seconds.

- **icmp-timeout** - Timeout value for ICMP packets.

  The timeout value ranges from 60 seconds to 172800 seconds. The default value is 3600 seconds.

- **sampling-timeout** - Time after which the device checks for dynamic translation activity.

  The timeout value ranges from 900 seconds to 172800 seconds.

To configure the timer values, see Configuring FINRST and SYN Timers.

Note There are three different options that can be configured for aging:

- Time-out: This is applicable for all type of flows (both TCP and UDP).
- TCP TIME-OUT: This is applicable for only TCP flows.
- UDP TIME-OUT: This is applicable for only UDP flows.

The **udp-timeout** and the **timeout** value timers are triggered after the timeout configured for the **ip nat translation sampling-timeout** command expires.

After dynamic NAT translations are created, they must be cleared when not in use so that newer translations can be created, especially because the number of TCAM entries is limited

Note When you create dynamic entries without timeouts configured, they take the default timeout of one hour (60 minutes). If you enter the **clear ip nat translations all** command after configuring timeouts, the configured timeout take effect. A timeout can be configured from 60 to 172800 seconds.

NAT Inside and Outside Addresses

NAT inside refers to networks owned by an organization that must be translated. When NAT is configured, hosts within this network will have addresses in one space (known as the local address space) that will appear to those outside the network as being in another space (known as the global address space).

Similarly, NAT outside refers to those networks to which the stub network connects. They are not generally under the control of the organization. Hosts in outside networks can be subject to translation and can have local and global addresses.

NAT uses the following definitions:

- Local address—A local IP address that appears on the inside of a network.
- Global address—A global IP address that appears on the outside of a network.
- Inside local address—The IP address that is assigned to a host on the inside network. The address is probably not a legitimate IP address assigned by the Internet Network Information Center (InterNIC) or a service provider.
- Inside global address—A legitimate IP address (assigned by InterNIC or a service provider) that represents one or more inside local IP addresses to the outside world.

- Outside local address—The IP address of an outside host as it appears to the inside network. The address is not necessarily legitimate; it was allocated from the address space that is routable on the inside.
- Outside global address—The IP address that is assigned to a host on the outside network by the owner of the host. The address was allocated from a globally routable address or a network space.

Pool Support for Dynamic NAT

Cisco NX-OS provides pool support for dynamic NAT. Dynamic NAT allows the configuration of a pool of global addresses that can be used to dynamically allocate a global address from the pool for every new translation. The addresses are returned to the pool after the session ages out or is closed. This allows for a more efficient use of addresses based on requirements.

Support for PAT includes the use of the global address pool. This further optimizes IP address utilization. PAT exhausts one IP address at a time with the use of port numbers. If no port is available from the appropriate group and more than one IP address is configured, PAT moves to the next IP address and gets the allocation based on the user defined pool (ignoring the source port or attempting to preserve it).

With dynamic NAT and PAT, each host uses a different address or port for each subsequent translation. The main difference between dynamic NAT and static NAT is that static NAT allows a remote host to initiate a connection to a translated host if an access list exists that allows it, while dynamic NAT does not.

When dynamic NAT is configured to use a pool of IP addresses, that are not locally available or configured locally, the out-to-in traffic is considered as DEST MISS. Due to this behavior, the **`show system internal access-list dest-miss stats`** command output displays increment in DEST MISS counters. The DEST MISS statistics is supported from Cisco NX-OS Release 9.3(5) onwards.

Static and Dynamic Twice NAT Overview

When both the source IP address and the destination IP address are translated as a single packet that goes through a Network Address Translation (NAT) device, it is referred to as twice NAT. Twice NAT is supported for static and dynamic translations.

Twice NAT allows you to configure two NAT translations (one inside and one outside) as part of a group of translations. These translations can be applied to a single packet as it flows through a NAT device. When you add two translations as part of a group, both the individual translations and the combined translation take effect.

A NAT inside translation modifies the source IP address and port number when a packet flows from inside to outside. It modifies the destination IP address and port number when the packet returns from outside to inside. NAT outside translation modifies the source IP address and port number when the packet flows from outside to inside, and it modifies the destination IP address and port number when the packet returns from inside to outside.

Without twice NAT, only one of the translation rules is applied on a packet, either the source IP address and port number or the destination IP address and port number.

Static NAT translations that belong to the same group are considered for twice NAT configuration. If a static configuration does not have a configured group ID, the twice NAT configuration will not work. All inside and outside NAT translations that belong to a single group that is identified by the group ID are paired to form twice NAT translations.

Dynamic twice NAT translations dynamically select the source IP address and port number information from pre-defined **ip nat pool** or **interface overload** configurations. Packet filtration is done by configuring ACLs, and traffic must originate from the dynamic NAT translation rule direction such that source translation is done by using dynamic NAT rules.

Dynamic twice NAT allows you to configure two NAT translations (one inside and one outside) as part of a group of translations. One translation must be dynamic and other translation must be static. When these two translations are part of a group of translations, both the translations can be applied on a single packet as it goes through the NAT device either from inside to outside or from outside to inside.

VRF Aware NAT

The VRF aware NAT feature enables a switch to understand an address space in a VRF (virtual routing and forwarding instances) and to translate the packet. This allows the NAT feature to translate traffic in an overlapping address space that is used between two VRFs.

Notes for VRF aware NAT:

- VRF over NAT is supported on 9300-FX3 platform switches.
- The VRF aware NAT feature is supported on N9K-9408PC-CFP2, N9K-X9564PX, N9K-C9272Q, N9K-C9272Q, N9K-X9464TX, N9K-X9464TX2, N9K-X9564TX, N9K-X9464PX, N9K-X9536PQ, N9K-X9636PQ, N9K-X9432PQ, N9K-C9332PQ, N9K-C9372PX, N9K-C9372PX-E, N9K-C9372TX, N9K-C9372TX-E, N9K-C93120TX.
- The VRF aware NAT feature is not supported on the Cisco Nexus 9300-EX platform switches.

Note This is a NAT TCAM limitation for the Cisco Nexus 9300-EX platform switches. NAT TCAM is not VRF aware. NAT does not work with overlapping IP addresses on Cisco Nexus 9300-EX platform switches.

- Beginning with Cisco NX-OS Release 10.2(3)F, VRF aware NAT is supported on Cisco Nexus 9300-FX, FX2, GX and GX2 platform switches. It is not supported on Cisco Nexus 9346C switch.
- Traffic flowing from one non-default-vrf to another non-default-vrf is not translated. (For example, vrfA to vrfB.)
- For traffic flowing from a VRF to a global-VRF, a nat-outside configuration is not supported on a non-default VRF interface.
- VRF aware NAT is supported by static and dynamic NAT configurations.
  - When traffic is configured to flow from a non-default VRF (inside) to a default VRF (outside), the **match-in-vrf** option of the **ip nat** command cannot be specified.
  - When traffic is configured to flow from a non-default VRF (inside) to the same non-default VRF (outside), the **match-in-vrf** option of the **ip nat** command must be specified.

    The following is an example configuration:

```
Switch(config)# ip nat inside source {list <acl-name>} {pool <pool-name> [vrf
<vrf-name> [match-in-vrf]] [overload] | interface <globalAddrInterface> [vrf
<vrf-name> [match-in-vrf]] overload} [group <group-id> dynamic]
```

```
Switch(config)#ip nat outside source list <acl-name> pool <pool-name> [vrf <vrf-name>
 [match-in-vrf]] [group <group-id> dynamic]}
```

- VRF aware NAT does not support fragmented packets.
- VRF aware NAT does not support application layer translations.

  Therefore, Layer 4 and other embedded IPs are not translated and the following will fail:

  - FTP
  - ICMP failures
  - IPSec
  - HTTPS

- VRF aware NAT supports NAT or VACL on an interface. (However, both features cannot be supported at the same time on an interface.)
- VRF aware NAT supports egress ACLs that are applied to the original packet, not on the NAT translated packet.
- VRF aware NAT supports only the default VRF.
- VRF aware NAT does not provide MIB support.
- VRF aware NAT does not provide DCNM support.
- VRF aware NAT supports only a single global VDC.
- VRF aware NAT does not support the active/standby supervisor model.
- VRFs with overlapping subnets cannot go to a common destination without NAT. However, you can achieve this functionality with inter-VRF NAT on dynamic NAT rule configuration. Static NAT configuration is not supported for overlapping address.

Guidelines and Limitations for Static NAT

Static NAT has the following configuration guidelines and limitations:

- For Broadcom-based Cisco Nexus 9000 Series switches, if the route to your inside global address on the translating device is reachable via the outside interface, packets for Network Address Translated flows coming from outside to inside get software forwarded, duplicated, and looped in the network. For this situation, you must enter the **add-route** CLI argument on the end of the NAT configuration for this flow. For example, **`ip nat inside source static 192.168.1.1 172.16.1.1 add-route`**.
- **show** commands with the **internal** keyword are not supported.
- NAT supports up to 1024 translations which include both static and dynamic NAT.
- If the translated IP is part of the outside interface subnet, then use the **ip proxy-arp** command on the NAT outside interface. Beginning with Cisco Nexus Release 9.2(1) and later , the nat-alias feature is enabled by default. You do not have to configure **ip proxy-arp** configuration.

- NAT and sFlow are not supported on the same port.
- The Cisco Nexus device supports NAT on the following interface types:
  - Switch Virtual Interfaces (SVIs)
  - Routed ports
  - Layer 3 and Layer 3 subinterfaces.
- NAT is supported on the default Virtual Routing and Forwarding (VRF) table only.
- NAT is supported for IPv4 Unicast only.
- The Cisco Nexus device does not support the following:
  - Software translation. All translations are done in the hardware.
  - Application layer translation. Layer 4 and other embedded IPs are not translated, including FTP, ICMP failures, IPSec, and HTTPs.
  - NAT and VLAN Access Control Lists (VACLs) that are configured on an interface at the same time.
  - PAT translation of fragmented IP packets.
  - NAT translation on software forwarded packets. For example, packets with IP-options are not NAT translated.
- By default no TCAM entries are allocated for the NAT feature. You allocate the TCAM size for the NAT feature by adjusting the TCAM size of other features. The TCAM can be allocated with the **hardware access-list tcam region nat** *tcam-size* command.
- HSRP and VRRP are not supported on a NAT interface.
- If an IP address is used for Static NAT or PAT translations, it cannot be used for any other purpose. For example, it cannot be assigned to an interface.
- For Static NAT, the outside global IP address should be different from the outside interface IP address.
- When configuring a large number of translations (more than 100), it is faster to configure the translations before configuring the NAT interfaces.
- UDF-based features may not work when NAT TCAM is carved.
- ECMP NAT is not supported on Cisco Nexus 9000 switches.
- NAT configurations such as **ip nat inside** or **ip nat outside** are not supported on loopback interfaces.

Restrictions for Dynamic NAT

The following restrictions apply to dynamic Network Address Translation (NAT):

- For Broadcom-based Cisco Nexus 9000 Series switches, if the route to your inside global address on the translating device is reachable via the outside interface, packets for Network Address Translated flows coming from outside to inside get software forwarded, duplicated, and looped in the network. For this

situation, you must enter the **add-route** CLI argument on the end of the NAT configuration for this flow. For example, `ip nat inside source static 192.168.1.1 172.16.1.1 add-route`.

- **show** commands with the **internal** keyword are not supported.
- The **interface overload option for inside policies** option is not supported on the Cisco Nexus 9200, 9300-EX, 9300-FX 9300-FX2, 9300-FX3, 9300-FXP, and 9300-GX platform switches for both outside and inside policies.
- VXLAN routing is not supported on Cisco Nexus devices.
- Fragmented packets are not supported.
- Application layer gateway (ALG) translations are not supported. ALG, also known as application-level gateway, is an application that translates IP address information inside the payload of an application packet.
- Egress ACLs are not applied to translated packets.
- Nondefault virtual routing and forwarding (VRF) instances are not supported.
- MIBs are not supported.
- Cisco Data Center Network Manager (DCNM) is not supported.
- Multiple global virtual device contexts (VDCs) are not supported on Cisco Nexus devices.
- Dynamic NAT translations are not synchronized with active and standby devices.
- Stateful NAT is not supported. However, NAT and Hot Standby Router Protocol (HSRP) can coexist.
- The timeout value for take up to the configured time-out + 119 seconds.
- Normally, ICMP NAT flows time out after the expiration of the configured sampling-timeout and translation-timeout. However, when ICMP NAT flows present in the switch become idle, they time out immediately after the expiration of the sampling-timeout configured.
- Hardware programming is introduced for ICMP on Cisco Nexus 9300 platform switches. Therefore, the ICMP entries consume the TCAM resources in the hardware. Because ICMP is in the hardware, the maximum limit for NAT translation in Cisco Nexus platform Series switches is changed to 1024. Maximum of 100 ICMP entries are allowed to make the best usage of the resources.
- When creating a new translation on a Cisco Nexus 9000 Series switch, the flow is software forwarded until the translation is programmed in the hardware, which might take a few seconds. During this period, there is no translation entry for the inside global address. Therefore, returning traffic is dropped. To overcome this limitation, create a loopback interface and give it an IP address that belongs to the NAT pool.
- For dynamic NAT, pool overload and interface overload are not supported for the outside NAT.
- Because the NAT overload uses PBR (Policy-Based Routing), the maximum number of available next-hop entries in the PBR table determines NAT scale. If the number of NAT inside interfaces are within the range of available next-hops entries in the PBR table, the maximum NAT translation scale remains same. Otherwise, the maximum number of supported translations may reduce. PBR and NAT-overload are not mutually exclusive; they are mutually limiting.
- The Cisco Nexus devices does not support NAT and VLAN Access Control Lists (VACLs) that are configured on an interface at the same time.

- NAT configurations such as **ip nat inside** or **ip nat outside** are not supported on loopback interfaces.
- The dynamic NAT feature over vPC is not supported.
- If traffic ingresses a PBR enabled interface, and has a NAT entry, the traffic will be routed via PBR but the IP address will not be translated.

Guidelines and Limitations for Dynamic Twice NAT

For Broadcom-based Cisco Nexus 9000 Series switches, if the route to your inside global address on the translating device is reachable via the outside interface, packets for Network Address Translated flows coming from outside to inside get software forwarded, duplicated, and looped in the network. For this situation, you must enter the **add-route** CLI argument on the end of the NAT configuration for this flow. For example, **`ip nat inside source static 192.168.1.1 172.16.1.1 add-route`**.

IP packets without TCP/UDP/ICMP headers are not translated with dynamic NAT.

In dynamic twice NAT, if dynamic NAT flows are not created before creating static NAT flows, dynamic twice NAT flows are not created correctly.

When an empty ACL is created, the default rule of **permit ip any any** is configured. The NAT-ACL does not match further ACL entries if the first ACL is blank.

Guidelines and Limitations for TCP Aware NAT

TCP aware NAT has the following limitations:

- TCP aware NAT is *not* supported on Cisco Nexus 9500 series switches.

  TCP aware NAT is supported on Cisco Nexus 9300-EX, FX, and FX2 series switches.
- Beginning with Cisco NX-OS Release 9.3(5), TCP aware NAT is supported on Cisco Nexus N9K-C9316D-GX, N9K-C93600CD-GX, N9K-C9364C-GX switches.
- Only one match ACL can be associated with one range of addresses pool. After associating a pool to a match ACL you cannot change the interface IP or modify the pool range.
- You must define the pool before configuring or using it in a dynamic NAT configuration.
- The dynamic NAT rule must be reconfigured whenever there is a change in pool range or interface address in case of interface overload.

Configuring Static NAT

Enabling Static NAT

SUMMARY STEPS

1. switch# **configure terminal**

2. switch(config)# **feature nat**
3. switch(config)# **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **feature nat**	Enables the static NAT feature on the device.
Step 3	switch(config)# **copy running-config startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Configuring Static NAT on an Interface

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **interface** *type slot*/*port*
3. switch(config-if)# **ip nat** {**inside** | **outside**}
4. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface** *type slot*/*port*	Specifies an interface to configure, and enters interface configuration mode.
Step 3	switch(config-if)# **ip nat** {**inside** | **outside**}	Specifies the interface as inside or outside. **Note** Only packets that arrive on a marked interface can be translated. This configuration is not supported on loopback interface.
Step 4	(Optional) switch(config)# **copy running-config startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

This example shows how to configure an interface with static NAT from the inside:

```
switch# configure terminal
switch(config)# interface ethernet 1/4
switch(config-if)# ip nat inside
```

Enabling Static NAT for an Inside Source Address

For inside source translation, the traffic flows from inside interface to the outside interface. NAT translates the inside local IP address to the inside global IP address. On the return traffic, the destination inside global IP address gets translated back to the inside local IP address.

Note When the Cisco Nexus device is configured to translate an inside source IP address (Src:ip1) to an outside source IP address (newSrc:ip2), the Cisco Nexus device implicitly adds a translation for an outside destination IP address (Dst: ip2) to an inside destination IP address (newDst: ip1).

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip nat inside source static** *local-ip-address global-ip-address* [**vrf** *vrf-name*] [**match-in-vrf**] [**group** *group-id*]
3. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **ip nat inside source static** *local-ip-address global-ip-address* [**vrf** *vrf-name*] [**match-in-vrf**] [**group** *group-id*]	Configures static NAT to translate the inside local address to the inside global address or to translate the opposite (the inside global traffic to the inside local traffic). Specifying **group** specifies the group to which this translation belongs on the static twice NAT. **Note** While performing twice NAT configuration in Cisco Nexus 9000 Series switches, you cannot use the same group ID across different VRFs. A unique group ID should be used for unique twice NAT rules.

	Command or Action	Purpose
Step 3	(Optional) switch(config)# **copy running-config startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

This example shows how to configure static NAT for an inside source address:

```
switch# configure terminal
switch(config)# ip nat inside source static 1.1.1.1 5.5.5.5
switch(config)# copy running-config startup-config
```

Enabling Static NAT for an Outside Source Address

For outside source translation, the traffic flows from the outside interface to the inside interface. NAT translates the outside global IP address to the outside local IP address. On the return traffic, the destination outside local IP address gets translated back to outside global IP address.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip nat outside source static** *outsideGlobalIP outsideLocalIP* [**vrf** *vrf-name* [**match-in-vrf**] [**group** *group-id*] [**dynamic**] [**add-route**]]
3. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **ip nat outside source static** *outsideGlobalIP outsideLocalIP* [**vrf** *vrf-name* [**match-in-vrf**] [**group** *group-id*] [**dynamic**] [**add-route**]]	Configures static NAT to translate the outside global address to the outside local address or to translate the opposite (the outside local traffic to the outside global traffic). Specifying **group** specifies the group to which this translation belongs on the static twice NAT. When an inside translation without ports is configured, an implicit add route is performed. The original add route functionality is an option while configuring an outside translation.
Step 3	(Optional) switch(config)# **copy running-config startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

This example show how to configure static NAT for an outside source address:

```
switch# configure terminal
switch(config)# ip nat outside source static 2.2.2.2 6.6.6.6
switch(config)# copy running-config startup-config
```

Configuring Static PAT for an Inside Source Address

You can map services to specific inside hosts using Port Address Translation (PAT).

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip nat inside source static** {*inside-local-address inside-global-address* | {**tcp**| **udp**} *inside-local-address* {*local-tcp-port* | *local-udp-port*} *inside-global-address* {*global-tcp-port* | *global-udp-port*}} {**vrf** *vrf-name* {**match-in-vrf**} {**group** *group-id*} }
3. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **ip nat inside source static** {*inside-local-address inside-global-address* | {**tcp**| **udp**} *inside-local-address* {*local-tcp-port* | *local-udp-port*} *inside-global-address* {*global-tcp-port* | *global-udp-port*}} {**vrf** *vrf-name* {**match-in-vrf**} {**group** *group-id*} }	Maps static NAT to an inside local port to an inside global port.
Step 3	(Optional) switch(config)# **copy running-config startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

This example shows how to map UDP services to a specific inside source address and UDP port:

```
switch# configure terminal
switch(config)#  ip nat inside source static udp 20.1.9.2 63 35.48.35.48 130
switch(config)# copy running-config startup-config
```

Configuring Static PAT for an Outside Source Address

You can map services to specific outside hosts using Port Address Translation (PAT).

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip nat outside source static** {*outside-global-address outside-local-address* | {**tcp** | **udp**} *outside-global-address* {*global-tcp-port* | *global-udp-port*} *outside-local-address* {*global-tcp-port* | *global-udp-port*}} {**group** *group-id*} {**add-route**} {**vrf** *vrf-name* {**match-in-vrf**}}
3. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **ip nat outside source static** {*outside-global-address outside-local-address* | {**tcp** | **udp**} *outside-global-address* {*global-tcp-port* | *global-udp-port*} *outside-local-address* {*global-tcp-port* | *global-udp-port*}} {**group** *group-id*} {**add-route**} {**vrf** *vrf-name* {**match-in-vrf**}}	Maps static NAT to an outside global port to an outside local port. Specifying **group** specifies the group to which this translation belongs on the static twice NAT. When an inside translation without ports is configured, an implicit add route is performed. The original add route functionality is an option while configuration an outside translation.
Step 3	(Optional) switch(config)# **copy running-config startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

This example shows how to map TCP services to a specific outside source address and TCP port:

```
switch# configure terminal
switch(config)#  ip nat outside source static tcp 20.1.9.2 63 35.48.35.48 130
switch(config)# copy running-config startup-config
```

Configuring Static Twice NAT

All translations within the same group are considered for creating static twice Network Address Translation (NAT) rules.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nat inside source static** *inside-local-ip-address inside-global-ip-address* [**group** *group-id*] [**add-route**]
4. **ip nat outside source static** *outside-global-ip-address outside-local-ip-address* [**group** *group-id*] [**add-route**]
5. **interface** *type number*

6. **ip address** *ip-address mask*
7. **ip nat inside**
8. **exit**
9. **interface** *type number*
10. **ip address** *ip-address mask*
11. **ip nat outside**
12. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `switch> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `switch# configure terminal`	Enters privileged EXEC mode.
Step 3	**ip nat inside source static** *inside-local-ip-address inside-global-ip-address* [**group** *group-id*] [**add-route**] **Example:** `switch(config)# ip nat inside source static 10.1.1.1 192.168.34.4 group 4`	Configures static twice NAT to translate an inside local IP address to the corresponding inside global IP address. • The **group** keyword determines the group to which a translation belongs.
Step 4	**ip nat outside source static** *outside-global-ip-address outside-local-ip-address* [**group** *group-id*] [**add-route**] **Example:** `switch(config)# ip nat outside source static 209.165.201.1 10.3.2.42 group 4 add-route`	Configures static twice NAT to translate an outside global IP address to the corresponding outside local IP address. • The **group** keyword determines the group to which a translation belongs.
Step 5	**interface** *type number* **Example:** `switch(config)# interface ethernet 1/2`	Configures an interface and enters interface configuration mode.
Step 6	**ip address** *ip-address mask* **Example:** `switch(config-if)# ip address 10.2.4.1 255.255.255.0`	Sets a primary IP address for an interface.
Step 7	**ip nat inside** **Example:** `switch(config-if)# ip nat inside`	Connects the interface to an inside network, which is subject to NAT. **Note** Configuration not supported on loopback interface.

	Command or Action	Purpose
Step 8	**exit** **Example:** `switch(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 9	**interface** *type number* **Example:** `switch(config)# interface ethernet 1/1`	Configures an interface and enters interface configuration mode.
Step 10	**ip address** *ip-address mask* **Example:** `switch(config-if)# ip address 10.5.7.9 255.255.255.0`	Sets a primary IP address for an interface.
Step 11	**ip nat outside** **Example:** `switch(config-if)# ip nat outside`	Connects the interface to an outside network, which is subject to NAT. **Note** Configuration not supported on loopback interface.
Step 12	**end** **Example:** `switch(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Enabling and Disabling no-alias Configuration

NAT devices own Inside Global (IG) and Outside Local (OL) addresses and they are responsible for responding to any ARP requests directed to these addresses. When the IG/OL address subnet matches with the local interface subnet, NAT installs an IP alias and an ARP entry, in this case the device uses local-proxy-arp to respond to ARP requests.

The *no-alias* feature responds to ARP requests of all the translated IPs from a given NAT pool address range if the address range is in same subnet of the outside interface.

If no-alias is enabled on an interface with NAT configuration, the outside interface will not respond to any ARP requests in its subnet. When no-alias is disabled, the ARP requests for IPs in same subnet as of outside interface are served.

Note When you downgrade to any older releases that does not support this feature, configurations with *no-alias* option may be deleted.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **feature nat**
3. switch(config)# **show run nat**

4. switch(config)# **show ip nat-alias**
5. switch(config)# **clear ip nat-alias** *ip address*/*all*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **feature nat**	Enables the static NAT feature on the device.
Step 3	switch(config)# **show run nat**	Displays NAT configuration.
Step 4	switch(config)# **show ip nat-alias**	Displays the information whether or not the alias is created. **Note** By default, alias is created. To disable the alias, you must append *no-alias* keyword to the command.
Step 5	switch(config)# **clear ip nat-alias** *ip address*/*all*	Removes entries from alias list. To remove a specific entry you must provide the IP address that you want to remove. To remove all entries, use the all keyword.

Example

This example shows the interface information:

```
switch# configure terminal
switch(config)# show ip int b
IP Interface Status for VRF "default"(1)
Interface            IP Address      Interface Status
Lo0                  100.1.1.1       protocol-up/link-up/admin-up
Eth1/1               7.7.7.1         protocol-up/link-up/admin-up
Eth1/3               8.8.8.1         protocol-up/link-up/admin-up
```

This example shows the running configuration:

```
switch# configure terminal
switch(config)# show running-config nat
!Command: show running-config nat
!Running configuration last done at: Thu Aug 23 11:57:01 2018
!Time: Thu Aug 23 11:58:13 2018

version 9.2(2) Bios:version 07.64
feature nat
interface Ethernet1/1
  ip nat inside
interface Ethernet1/3
  ip nat outside
switch(config)#
```

This example shows how to configure alias:

```
switch# configure terminal
switch(config)# ip nat pool p1 7.7.7.2 7.7.7.20 prefix-length 24
```

```
switch(config)# ip nat inside source static 1.1.1.2 8.8.8.3
switch(config)# ip nat outside source static 2.2.2.1 7.7.7.3
switch(config)# show ip nat-alias
Alias Information for Context: default
Address         Interface
7.7.7.2          Ethernet1/1
8.8.8.2          Ethernet1/3
switch(config)#
```

This example shows the output of *show ip nat-alias*. By default, alias is enabled.

```
switch# configure terminal
switch(config)# show ip nat-alias
Alias Information for Context: default
Address         Interface
7.7.7.2          Ethernet1/1
8.8.8.2          Ethernet1/3
switch(config)#
```

This example shows how to disable alias:

```
switch# configure terminal
switch(config)# ip nat pool p1 7.7.7.2 7.7.7.20 prefix-length 24 no-alias
switch(config)# ip nat inside source static 1.1.1.2 8.8.8.3 no-alias
switch(config)# ip nat outside source static 2.2.2.1 7.7.7.3 no-alias
switch(config)# show ip nat-alias
Alias Information for Context: default
Address         Interface
7.7.7.2          Ethernet1/1
8.8.8.2          Ethernet1/3
switch(config)#

** None of the entry got appended as alias is disabled for above CLIs.
switch(config)#
```

This example shows how to clear alias. Use *clear ip nat-alias* to remove an entry from alias list. You can remove a single entry by specifying the IP address or remove all the alias entries.

```
switch# configure terminal
switch(config)# clear ip nat-alias address 7.7.7.2
switch(config)# show ip nat-alias
Alias Information for Context: default
Address         Interface
8.8.8.2          Ethernet1/3
switch(config)#
switch(config)# clear ip nat-alias all
switch(config)# show ip nat-alias
switch(config)#
```

Configuration Example for Static NAT and PAT

This example shows the configuration for static NAT:

```
ip nat inside source static 103.1.1.1 11.3.1.1
ip nat inside source static 139.1.1.1 11.39.1.1
ip nat inside source static 141.1.1.1 11.41.1.1
ip nat inside source static 149.1.1.1 95.1.1.1
ip nat inside source static 149.2.1.1 96.1.1.1
ip nat outside source static 95.3.1.1 95.4.1.1
ip nat outside source static 96.3.1.1 96.4.1.1
ip nat outside source static 102.1.2.1 51.1.2.1
```

```
ip nat outside source static 104.1.1.1 51.3.1.1
ip nat outside source static 140.1.1.1 51.40.1.1
```

This example shows the configuration for static PAT:

```
ip nat inside source static tcp 10.11.1.1 1 210.11.1.1 101
ip nat inside source static tcp 10.11.1.1 2 210.11.1.1 201
ip nat inside source static tcp 10.11.1.1 3 210.11.1.1 301
ip nat inside source static tcp 10.11.1.1 4 210.11.1.1 401
ip nat inside source static tcp 10.11.1.1 5 210.11.1.1 501
ip nat inside source static tcp 10.11.1.1 6 210.11.1.1 601
ip nat inside source static tcp 10.11.1.1 7 210.11.1.1 701
ip nat inside source static tcp 10.11.1.1 8 210.11.1.1 801
ip nat inside source static tcp 10.11.1.1 9 210.11.1.1 901
ip nat inside source static tcp 10.11.1.1 10 210.11.1.1 1001
ip nat inside source static tcp 10.11.1.1 11 210.11.1.1 1101
ip nat inside source static tcp 10.11.1.1 12 210.11.1.1 1201
```

Example: Configuring Static Twice NAT

The following example shows how to configure the inside source and outside source static twice NAT configurations:

```
Switch> enable
Switch# configure terminal
Switch(config)# ip nat inside source static 10.1.1.1 192.168.34.4 group 4
Switch(config)# ip nat outside source static 209.165.201.1 10.3.2.42 group 4
Switch(config)# interface ethernet 1/2
Switch(config-if)# ip address 10.2.4.1 255.255.255.0
Switch(config-if)# ip nat inside
switch(config-if)# exit
switch(config)# interface ethernet 1/1
switch(config-if)# ip address 10.5.7.9 255.255.255.0
switch(config-if)# ip nat outside
Switch(config-if)# end
```

Verifying the Static NAT Configuration

To display the static NAT configuration, perform this task:

SUMMARY STEPS

1. switch# **show ip nat translations**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **show ip nat translations**	Shows the translations for the inside global, inside local, outside local, and outside global IP addresses.

Example

This example shows how to display the static NAT configuration:

```
switch# sh ip nat translations
Pro Inside global      Inside local       Outside local      Outside global

--- ---                ---                51.3.1.1           104.1.1.1
--- ---                ---                95.4.1.1           95.3.1.1
--- ---                ---                96.4.1.1           96.3.1.1
--- ---                ---                51.40.1.1          140.1.1.1
--- ---                ---                51.42.1.1          142.1.2.1
--- ---                ---                51.1.2.1           102.1.2.1
--- 11.1.1.1           101.1.1.1          ---                ---
--- 11.3.1.1           103.1.1.1          ---                ---
--- 11.39.1.1          139.1.1.1          ---                ---
--- 11.41.1.1          141.1.1.1          ---                ---
--- 95.1.1.1           149.1.1.1          ---                ---
--- 96.1.1.1           149.2.1.1          ---                ---
    130.1.1.1:590      30.1.1.100:5000    ---                ---
    130.2.1.1:590      30.2.1.100:5000    ---                ---
    130.3.1.1:590      30.3.1.100:5000    ---                ---
    130.4.1.1:590      30.4.1.100:5000    ---                ---
    130.1.1.1:591      30.1.1.101:5000    ---                ---


switch# sh ip nat translations verbose
Pro Inside global      Inside local       Outside local      Outside global
any ---                ---                22.1.1.3           22.1.1.2
    Flags:0x200009 time-left(secs):-1 id:0 state:0x0 grp_id:10
any 11.1.1.130         11.1.1.3           ---                ---
    Flags:0x1 time-left(secs):-1 id:0 state:0x0 grp_id:0
any 11.1.1.133         11.1.1.33          ---                ---
    Flags:0x1 time-left(secs):-1 id:0 state:0x0 grp_id:10
any 11.1.1.133         11.1.1.33          22.1.1.3           22.1.1.2
    Flags:0x200009 time-left(secs):-1 id:0 state:0x0 grp_id:0
tcp 10.1.1.100:64490   10.1.1.2:0         20.1.1.2:0         20.1.1.2:0
    Flags:0x82 time-left(secs):43192 id:31 state:0x3 grp_id:0 vrf: default
N9300-1#
```

Configuring Dynamic NAT

Configuring Dynamic Translation and Translation Timeouts

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list** *access-list-name*
4. **permit** *protocol source source-wildcard* **any**
5. **deny** *protocol source source-wildcard* **any**
6. **exit**

7. **ip nat inside source list** *access-list-name* **interface** *type number* [**vrf** *vrf-name* [**match-in-vrf**] [**add-route**] [**overload**]
8. **interface** *type number*
9. **ip address** *ip-address mask*
10. **ip nat inside**
11. **exit**
12. **interface** *type number*
13. **ip address** *ip-address mask*
14. **ip nat outside**
15. **exit**
16. **ip nat translation max-entries** *number-of-entries*
17. **ip nat translation timeout** *seconds*
18. **ip nat translation creation-delay** *seconds*
19. **ip nat translation icmp-timeout** *seconds*
20. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Switch> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Switch# configure terminal`	Enters global configuration mode.
Step 3	**ip access-list** *access-list-name* **Example:** `Switch(config)# ip access-list acl1`	Defines an access list and enters access-list configuration mode.
Step 4	**permit** *protocol source source-wildcard* **any** **Example:** `Switch(config-acl)# permit ip 10.111.11.0/24 any`	Sets conditions in an IP access list that permit traffic matching the conditions.
Step 5	**deny** *protocol source source-wildcard* **any** **Example:** `Switch(config-acl)# deny udp 10.111.11.100/32 any`	Sets conditions in an IP access list that deny packets from entering a network.
Step 6	**exit** **Example:** `Switch(config-acl)# exit`	Exits access-list configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 7	**ip nat inside source list** *access-list-name* **interface** *type number* [**vrf** *vrf-name* [**match-in-vrf**] [**add-route**] [**overload**] **Example:** `Switch(config)# ip nat inside source list acl1 interface ethernet 1/1 overload`	Establishes dynamic source translation by specifying the access list defined in Step 3.
Step 8	**interface** *type number* **Example:** `Switch(config)# interface ethernet 1/4`	Configures an interface and enters interface configuration mode.
Step 9	**ip address** *ip-address mask* **Example:** `Switch(config-if)# ip address 10.111.11.39 255.255.255.0`	Sets a primary IP address for the interface.
Step 10	**ip nat inside** **Example:** `Switch(config-if)# ip nat inside`	Connects the interface to an inside network, which is subject to NAT. **Note** Configuration not supported on loopback interface.
Step 11	**exit** **Example:** `Switch(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 12	**interface** *type number* **Example:** `Switch(config)# interface ethernet 1/1`	Configures an interface and enters interface configuration mode.
Step 13	**ip address** *ip-address mask* **Example:** `Switch(config-if)# ip address 172.16.232.182 255.255.255.240`	Sets a primary IP address for an interface.
Step 14	**ip nat outside** **Example:** `Switch(config-if)# ip nat outside`	Connects the interface to an outside network. **Note** Configuration not supported on loopback interface.
Step 15	**exit** **Example:** `Switch(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 16	**ip nat translation max-entries** *number-of-entries* **Example:** `Switch(config)# ip nat translation max-entries 300`	Specifies the maximum number of dynamic NAT translations. The number of entries can be between 1 and 1023.
Step 17	**ip nat translation timeout** *seconds* **Example:** `switch(config)# ip nat translation timeout 13000`	Specifies the timeout value for dynamic NAT translations.
Step 18	**ip nat translation creation-delay** *seconds* **Example:** `switch(config)# ip nat translation creation-delay 250`	Specifies the ICMP timeout value for dynamic NAT translations. **Note** To reduce the frequency of programming the NAT entries in the hardware, NAT batches and programs the translations for one second. Frequently programming the hardware burdens the CPU but delaying the programming delays establishing sessions. You can disable batching or reduce the creation delay using this command. It is not recommended to set creation delay to 0.
Step 19	**ip nat translation icmp-timeout** *seconds* **Example:** `switch(config)# ip nat translation icmp-timeout 100`	Specifies the ICMP timeout value for dynamic NAT translations.
Step 20	**end** **Example:** `Switch(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring Dynamic NAT Pool

You can create a NAT pool by either defining the range of IP addresses in a single **ip nat pool** command or by using the **ip nat pool** and **address** commands

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **feature nat**
3. switch(config)# **ip nat pool** *pool-name* [*startip endip*] {**prefix** *prefix-length* | **netmask** *network-mask*}
4. (Optional) switch(config-ipnat-pool)# **address** *startip endip*
5. (Optional) switch(config)# **no ip nat pool** *pool-name*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **feature nat**	Enables the NAT feature on the device.
Step 3	switch(config)# **ip nat pool** *pool-name* [*startip endip*] {**prefix** *prefix-length* | **netmask** *network-mask*}	Creates a NAT pool with a range of global IP addresses. The IP addresses are filtered by using either a prefix length or a network mask.
Step 4	(Optional) switch(config-ipnat-pool)# **address** *startip endip*	Specifies the range of global IP addresses if they were not specified during creation of the pool.
Step 5	(Optional) switch(config)# **no ip nat pool** *pool-name*	Deletes the specified NAT pool.

Example

This example shows how to create a NAT pool with a prefix length:

```
switch# configure terminal
switch(config)# ip nat pool pool1 30.1.1.1 30.1.1.2 prefix-length 24
switch(config)#
```

This example shows how to create a NAT pool with a network mask:

```
switch# configure terminal
switch(config)# ip nat pool pool5 20.1.1.1 20.1.1.5 netmask 255.0.255.0
switch(config)#
```

This example shows how to create a NAT pool and define the range of global IP addresses using the **ip nat pool** and **address** commands:

```
switch# configure terminal
switch(config)# ip nat pool pool7 netmask 255.255.0.0
switch(config-ipnat-pool)# address 40.1.1.1 40.1.1.5
switch(config-ipnat-pool)#
```

This example shows how to delete a NAT pool:

```
switch# configure terminal
switch(config)# no ip nat pool pool4
switch(config)#
```

Configuring Source Lists

You can configure a source list of IP addresses for the inside interface and the outside interface.

Before you begin

Ensure that you configure a pool before configuring the source list for the pool.

SUMMARY STEPS

1. switch# **configure terminal**
2. (Optional) switch# **ip nat inside source list** *list-name* **pool** *pool-name* [**overload**]
3. (Optional) switch# **ip nat outside source list** *list-name* **pool** *pool-name* [**add-route**]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	(Optional) switch# **ip nat inside source list** *list-name* **pool** *pool-name* [**overload**]	Creates a NAT inside source list with pool with or without overloading.
Step 3	(Optional) switch# **ip nat outside source list** *list-name* **pool** *pool-name* [**add-route**]	Creates a NAT outside source list with pool without overloading.

Example

This example shows how to create a NAT inside source list with pool without overloading:

```
switch# configure terminal
switch(config)# ip nat inside source list list1 pool pool1
switch(config)#
```

This example shows how to create a NAT inside source list with pool with overloading:

```
switch# configure terminal
switch(config)# ip nat inside source list list2 pool pool2 overload
switch(config)#
```

This example shows how to create a NAT outside source list with pool without overloading:

```
switch# configure terminal
switch(config)# ip nat outside source list list3 pool pool3
switch(config)#
```

Configuring Dynamic Twice NAT for an Inside Source Address

For an inside source address translation, the traffic flows from the inside interface to the outside interface. You can configure dynamic twice NAT for an inside source address.

Before you begin

Ensure that you enable NAT on the switch.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip nat outside source static** *outside-global-ip-address outside-local-ip-address* | [**tcp** | **udp**] *outside-global-ip-address outside-global-port outside-local-ip-address outside-local-port* [**group** *group-id*] [**dynamic**] [**add-route**]
3. switch(config)# **ip nat inside source list** *access-list-name* [**interface** *type slot/port* **overload** | **pool** *pool-name* **overload**] [**group** *group-id*] [**dynamic**] [**add-route**]
4. switch(config)# **ip nat pool** *pool-name* [*startip endip*] {**prefix** *prefix-length* | **netmask** *network-mask*}
5. switch(config)# **interface** *type slot/port*
6. switch(config-if)# **ip nat outside**
7. switch(config-if)# **exit**
8. switch(config)# **interface** *type slot/port*
9. switch(config-if)# **ip nat inside**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **ip nat outside source static** *outside-global-ip-address outside-local-ip-address* | [**tcp** | **udp**] *outside-global-ip-address outside-global-port outside-local-ip-address outside-local-port* [**group** *group-id*] [**dynamic**] [**add-route**]	Configures static NAT to translate an outside global address to an inside local address or to translate inside local traffic to inside global traffic. The **group** keyword determines the group to which a translation belongs.
Step 3	switch(config)# **ip nat inside source list** *access-list-name* [**interface** *type slot/port* **overload** | **pool** *pool-name* **overload**] [**group** *group-id*] [**dynamic**] [**add-route**]	Establishes dynamic source translation by creating a NAT inside source list with pool with or without overloading. The **group** keyword determines the group to which a translation belongs.
Step 4	switch(config)# **ip nat pool** *pool-name* [*startip endip*] {**prefix** *prefix-length* | **netmask** *network-mask*}	Creates a NAT pool with a range of global IP addresses. The IP addresses are filtered by using either a prefix length or a network mask.
Step 5	switch(config)# **interface** *type slot/port*	Configures an interface and enters interface configuration mode.
Step 6	switch(config-if)# **ip nat outside**	Connects the interface to an outside network.
Step 7	switch(config-if)# **exit**	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 8	switch(config)# **interface** *type slot/port*	Configures an interface and enters interface configuration mode.
Step 9	switch(config-if)# **ip nat inside**	Connects the interface to an inside network, which is subject to NAT.

Example

This example shows how to configure dynamic twice NAT for an inside source address:

```
switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)# ip nat outside source static 2.2.2.2 4.4.4.4 group 20 dynamic
switch(config)# ip nat inside source list acl_1 pool pool_1 overload group 20 dynamic
switch(config)# ip nat pool pool_1 3.3.3.3 3.3.3.10 prefix-length 24
switch(config)# interface Ethernet1/8
switch(config-if)# ip nat outside
switch(config-if)# exit
switch(config)# interface Ethernet1/15
switch(config-if)# ip nat inside
```

Configuring Dynamic Twice NAT for an Outside Source Address

For an outside source address translation, the traffic flows from the outside interface to the inside interface. You can configure dynamic twice NAT for an outside source address.

Before you begin

Ensure that you enable NAT on the switch.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip nat inside source static** *inside-local-ip-address inside-global-ip-address* | [**tcp** | **udp**] *inside-local-ip-address local-port inside-global-ip-address global-port* [**group** *group-id*] [**dynamic**] [**add-route**]
3. switch(config)# **ip nat outside source list** *access-list-name* **pool** *pool-name* [**group** *group-id*] **dynamic** [**add-route**]
4. switch(config)# **ip nat pool** *pool-name* [*startip endip*] {**prefix** *prefix-length* | **netmask** *network-mask*}
5. switch(config)# **interface** *type slot/port*
6. switch(config-if)# **ip nat outside**
7. switch(config-if)# **exit**
8. switch(config)# **interface** *type slot/port*
9. switch(config-if)# **ip nat inside**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **ip nat inside source static** *inside-local-ip-address inside-global-ip-address* | [**tcp** | **udp**] *inside-local-ip-address local-port inside-global-ip-address global-port* [**group** *group-id*] [**dynamic**] [**add-route**]	Configures static NAT to translate an inside global address to an inside local address or to translate inside local traffic to inside global traffic. The **group** keyword determines the group to which a translation belongs.
Step 3	switch(config)# **ip nat outside source list** *access-list-name* **pool** *pool-name* [**group** *group-id*] **dynamic** [**add-route**]	Establishes dynamic source translation by creating a NAT outside source list with pool with or without overloading.
Step 4	switch(config)# **ip nat pool** *pool-name* [*startip endip*] {**prefix** *prefix-length* | **netmask** *network-mask*}	Creates a NAT pool with a range of global IP addresses. The IP addresses are filtered by using either a prefix length or a network mask.
Step 5	switch(config)# **interface** *type slot/port*	Configures an interface and enters interface configuration mode.
Step 6	switch(config-if)# **ip nat outside**	Connects the interface to an outside network.
Step 7	switch(config-if)# **exit**	Exits interface configuration mode and returns to global configuration mode.
Step 8	switch(config)# **interface** *type slot/port*	Configures an interface and enters interface configuration mode.
Step 9	switch(config-if)# **ip nat inside**	Connects the interface to an inside network, which is subject to NAT.

Example

This example shows how to configure dynamic twice NAT for an outside source address:

```
switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)# ip nat inside source static 7.7.7.7 5.5.5.5 group 30 dynamic
switch(config)# ip nat outside source list acl_1 pool pool_1 group 30 dynamic
switch(config)# ip nat pool pool_2 4.4.4.4 4.4.4.10 prefix-length 24
switch(config)# interface Ethernet1/6
switch(config-if)# ip nat outside
switch(config-if)# exit
switch(config)# interface Ethernet1/11
switch(config-if)# ip nat inside
```

Configuring FINRST and SYN Timers

This section describes how to configure FINRST and SYN timer values.

When you reload the switch, restoring or erasing the configured FINRST and/or SYN timer values depends on whether or not the TCP TCAM carved. If the TCAM is carved, the switch restores the currently configured values.

If the timer values are *not* configured, it sets a default value of 60 seconds. If the TCAM is *not* carved, the switch removes any currently configured values and sets a default value as never. This is because the the TCP AWARE feature gets disabled when the TCP TCAM is not carved.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config-if)# **ip nat translation syn-timeout** {**seconds** | **never**}
3. switch(config-if)# **ip nat translation finrst-timeout** {**seconds** | **never**}

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config-if)# **ip nat translation syn-timeout** {**seconds** | **never**}	Specifies the timeout value for TCP data packets that sends the SYN request, but do not receive a SYN-ACK reply. The timeout value ranges from 1 to 172800 seconds. When the TCP TCAM is carved the default value is 60 seconds. When the TCP TCAM is *not* carved the default value is *never*. The *never* keyword deactivates SYN timer. **Note** You cannot configure SYN timer when TCP TCAM is *not* carved..
Step 3	switch(config-if)# **ip nat translation finrst-timeout** {**seconds** | **never**}	Specifies the timeout value for the flow entries when a connection is terminated by receiving finish (FIN) or reset (RST) packets. You must use the configure the behavior for both RST and FIN. The timeout value ranges from 1 to 172800 seconds. When the TCP TCAM is carved the default value is 60 seconds. When the TCP TCAM is not carved the default value is *never*. The *never* keyword deactivates FIN or RST timers. **Note** You cannot configure FINRST timer if TCP TCAM is *not* carved..

Example

The following example that shows when TCP TCAM is carved

```
switch(config)# ip nat translation syn-timeout 20
```

The following example that shows when TCP TCAM is not carved

```
switch(config)# ip nat translation syn-timeout 20
Error: SYN TIMER CONFIG FAILED.TCP TCAM NOT CONFIGURED
```

Clearing Dynamic NAT Translations

To clear dynamic translations, perform the following task:

Command	Purpose
clear ip nat translation [**all** | **inside** *global-ip-address local-ip-address* [**outside** *local-ip-address global-ip-address*] | **outside** *local-ip-address global-ip-address*]	Deletes all or specific dynamic NAT translations.

Example

This example shows how to clear all dynamic translations:

```
switch# clear ip nat translation all
```

This example shows how to clear dynamic translations for inside and outside addresses:

```
switch# clear ip nat translation inside 2.2.2.2 4.4.4.4 outside 5.5.5.5 7.7.7.7
```

Verifying Dynamic NAT Configuration

To display dynamic NAT configuration, perform the following tasks:

Command	Purpose
show ip nat translations	Displays active Network Address Translation (NAT) translations. Displays additional information for each translation table entry, including when an entry was created and used.
show run nat	Displays NAT configuration.
show ip nat max	Displays active Network Address Translation (NAT) maximum values.
show ip nat statistics	Monitor NAT statistics.

Example

This example shows how to display IP NAT Max values:

```
switch# show ip nat max

IP NAT Max values
====================
Max Dyn Translations:80
Max all-host:0
No.Static:0
No.Dyn:1
No.Dyn-ICMP:1
====================
Switch(config)#
```

This example shows how to display NAT Statistics:

```
switch# show ip nat statistics

IP NAT Statistics
==================================================
Stats Collected since: Mon Feb 24 18:27:34 2020
--------------------------------------------------
Total active translations: 1
No.Static: 0
No.Dyn:    1
No.Dyn-ICMP:   1
--------------------------------------------------
Total expired Translations: 0
SYN timer expired:          0
FIN-RST timer expired:      0
Inactive timer expired:     0
--------------------------------------------------
Total Hits:  2                  Total Misses:  2
In-Out Hits: 0                  In-Out Misses: 2
Out-In Hits: 2                  Out-In Misses: 0
--------------------------------------------------
Total SW Translated Packets: 2
In-Out SW Translated: 2
Out-In SW Translated: 0
--------------------------------------------------
Total SW Dropped Packets: 0
In-Out SW Dropped: 0
Out-In SW Dropped: 0

Address alloc. failure drop:      0
Port alloc.  failure drop:        0
Dyn. Translation max limit drop:  0
ICMP max limit drop:              0
Allhost max limit drop:           0
--------------------------------------------------
Total TCP session established: 0
Total TCP session closed:      0
--------------------------------------------------
NAT Inside Interfaces:  1
Ethernet1/34

NAT Outside Interfaces: 1
Ethernet1/32
--------------------------------------------------
Inside source list:
```

```
++++++++++++++++++++

Access list: T2
RefCount: 1
Pool: T2     Overload
Total addresses: 10
Allocated: 1    percentage: 10%
Missed: 0


Outside source list:
++++++++++++++++++++
--------------------------------------------------
==================================================
Switch(config)#
Switch(config)#


**No.Dyn-ICMP  field is to display the no of icmp dynamic translations , its a subset of
"No.Dyn" field.
```

Note Beginning with Cisco NX-OS Release 9.3(5), the **No.Dyn-ICMP** field is a subset of **No.Dyn** field and it displays the number of ICMP dynamic translations.

This example shows how to display running configuration for NAT:

```
switch# show run nat

!Command: show running-config nat
!Time: Wed Apr 23 11:17:43 2014

version 6.0(2)A3(1)
feature nat

ip nat inside source list list1 pool pool1
ip nat inside source list list2 pool pool2 overload
ip nat inside source list list7 pool pool7 overload
ip nat outside source list list3 pool pool3
ip nat pool pool1 30.1.1.1 30.1.1.2 prefix-length 24
ip nat pool pool2 10.1.1.1 10.1.1.2 netmask 255.0.255.0
ip nat pool pool3 30.1.1.1 30.1.1.8 prefix-length 24
ip nat pool pool5 20.1.1.1 20.1.1.5 netmask 255.0.255.0
ip nat pool pool7 netmask 255.255.0.0
  address 40.1.1.1 40.1.1.5
```

This example shows how to display active NAT translations:

Inside pool with overload

```
switch# show ip nat translation
Pro  Inside global      Inside local       Outside local      Outside global
icmp 20.1.1.3:64762     10.1.1.2:133       20.1.1.1:0         20.1.1.1:0
icmp 20.1.1.3:64763     10.1.1.2:134       20.1.1.1:0         20.1.1.1:0
```

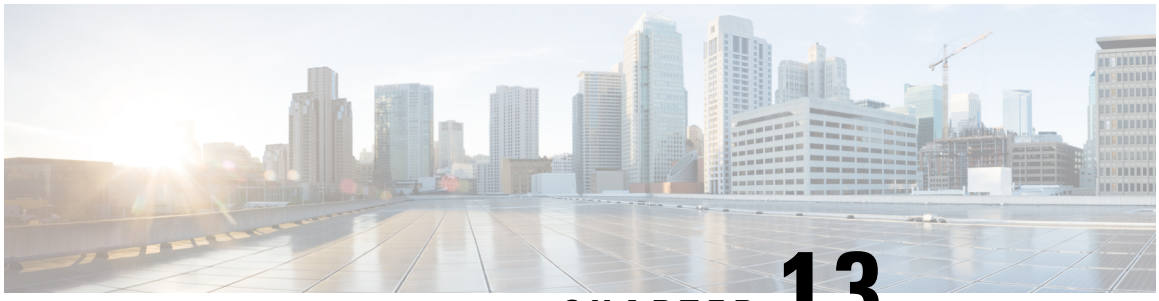
Outside pool without overload

```
switch# show ip nat translation
Pro  Inside global      Inside local       Outside local      Outside global
any  ---                ---                177.7.1.1:0        77.7.1.64:0
any  ---                ---                40.146.1.1:0       40.46.1.64:0
any  ---                ---                10.4.146.1:0       10.4.46.64:0
```

Example: Configuring Dynamic Translation and Translation Timeouts

The following example shows how to configure dynamic overload Network Address Translation (NAT) by specifying an access list:

```
Switch> enable
Switch# configure terminal
Switch(config)# ip access-list acl1
Switch(config-acl)# permit ip 10.111.11.0/24 any
Switch(config-acl)# deny udp 10.111.11.100/32 any
Switch(config-acl)# exit
Switch(config)# ip nat inside source list acl1 interface ethernet 1/1 overload
Switch(config)# interface ethernet 1/4
Switch(config-if)# ip address 10.111.11.39 255.255.255.0
Switch(config-if)# ip nat inside
Switch(config-if)# exit
Switch(config)# interface ethernet 1/1
Switch(config-if)# ip address 172.16.232.182 255.255.255.240
Switch(config-if)# ip nat outside
Switch(config-if)# exit
Switch(config)# ip nat translation max-entries 300
Switch(config)# ip nat translation timeout 13000
Switch(config)# end
```



CHAPTER 13

Configuring IP Event Dampening

IP Event Dampening Overview

Interface state changes occur when interfaces are administratively brought up or down or if an interface changes state. When an interface changes state or flaps, routing protocols are notified of the status of the routes that are affected by the change in state. Every interface state change requires all affected devices in the network to recalculate best paths, install or remove routes from the routing tables, and then advertise valid routes to peer routers. An unstable interface that flaps excessively can cause other devices in the network to consume substantial amounts of system processing resources and cause routing protocols to lose synchronization with the state of the flapping interface.

The IP Event Dampening feature introduces a configurable exponential decay mechanism to suppress the effects of excessive interface flapping events on routing protocols and routing tables in the network. This feature allows the network operator to configure a router to automatically identify and selectively dampen a local interface that is flapping. Dampening an interface removes the interface from the network until the interface stops flapping and becomes stable. Configuring the IP Event Dampening feature improves convergence times and stability throughout the network by isolating failures so that disturbances are not propagated. This, in turn, reduces the utilization of system processing resources by other devices in the network and improves overall network stability.

Guidelines and Limitations

The IP Event Dampening feature introduces a configurable exponential decay mechanism to suppress the effects of excessive interface flapping events on routing protocols and routing tables in the network. This feature allows the network operator to configure a router to automatically identify and selectively dampen a local interface that is flapping. See the following guidelines and limitations before configuring IP Event Dampening feature:

- Beginning from Cisco NX-OS Release 9.2(1), IP event dampening is supported on Cisco Nexus 9300-EX, 9300-FX, 9300-FX2, 9300-FXP, 9700-EX, and 9700-FX platform switches.

- Due to changes in the netstack-IP component, all the IP clients observe the impact of dampening or interface.
- For each flap of the interface, a certain penalty is added. The penalty decays exponentially whose parameters are configured.
- When penalty exceeds the Suppress threshold the interface is dampened. It is unsuppressed when the penalty decays below the Reuse threshold.
- When an interface is dampened, the IP address and the static routes are removed from the interface. All the clients of IP get an IP delete notification.
- When an interface is unsuppressed, the IP address and the relevant routes are added back. All the clients of IP get an IP address add notification for all the IP addresses of the interface.
- All Layer 3 interfaces that are configured on the Ethernet interface, port channels, and SVI support this feature.

Interface State Change Events

IP Event Dampening feature employs a configurable exponential decay mechanism that is used to suppress the effects of excessive interface flapping or state changes. When the IP Event Dampening feature is enabled, flapping interfaces are dampened from the perspective of the routing protocol by filtering excessive route updates. Flapping interfaces are identified, assigned penalties, suppressed if necessary, and made available to the network when the interface stabilizes.

Suppress Threshold

The suppress threshold is the value of the accumulated penalty that triggers the router to dampen a flapping interface. The flapping interface is identified by the router and assigned a penalty for each up and down state change, but the interface is not automatically dampened. The router tracks the penalties that a flapping interface accumulates. When the accumulated penalty reaches the default or preconfigured suppress threshold, the interface is placed in a dampened state.

Half-Life Period

The half-life period determines how fast the accumulated penalty can decay exponentially. When an interface is placed in a dampened state, the router monitors the interface for additional up and down state changes. If the interface continues to accumulate penalties and the interface remains in the suppress threshold range, the interface will remain dampened. If the interface stabilizes and stops flapping, the penalty is reduced by half after each half-life period expires. The accumulated penalty will be reduced until the penalty drops to the reuse threshold. The configurable range of the half-life period timer is from 1 to 30 seconds. The default half-life period timer is 5 seconds.

Reuse Threshold

When the accumulated penalty decreases until the penalty drops to the reuse threshold, the route is unsuppressed and made available to other devices in the network. The range of the reuse value is from 1 to 20000 penalties. The default value is 1000 penalties.

Maximum Suppress Time

The maximum suppress time represents the maximum time an interface can remain dampened when a penalty is assigned to an interface. The maximum suppress time can be configured from 1 to 255 seconds. The maximum penalty is truncated to maximum 20000 unit. The maximum value of the accumulated penalty is calculated based on the maximum suppress time, reuse threshold, and half-life period.

IP event dampening configuration command applies dampening to routing protocols for both IP and CLNS.

The first set of parameters ([half-life | reuse | suppress max-suppress]) configure the different parameters of the dampening algorithm. The second set ([restart [penalty]]) enables dampening penalty to be applied when the interface comes up the first time after reboot. The default restart penalty is applied only if you specify the restart parameter. Both parameter sets are optional

Affected Components

When an interface is not configured with dampening, or when an interface is configured with dampening but is not suppressed, the routing protocol behavior as a result of interface state transitions is not changed by the IP Event Dampening feature. However, if an interface is suppressed, the routing protocols and routing tables are immune to any further state transitions of the interface until it is unsuppressed.

Route Types

- Connected routes:
  - The connected routes of dampened interfaces are not installed into the routing table.
  - When a dampened interface is unsuppressed, the connected routes will be installed into the routing table if the interface is up.
- Static routes:
  - Static routes assigned to a dampened interface are not installed into the routing table.
  - When a dampened interface is unsuppressed, the static route will be installed into the routing table if the interface is up.

Note Only the primary interface can be configured with this feature, and all subinterfaces are subject to the same dampening configuration as the primary interface. IP Event Dampening does not track the flapping of individual subinterfaces on an interface.

Supported Protocols

All the protocols that are used are impacted by the IP Event Dampening feature. The IP Event Dampening feature supports Border Gateway Protocol (BGP), Enhanced Interior Gateway Routing Protocol (EIGRP), Hot Standby Routing Protocol (HSRP), Open Shortest Path First (OSPF), Routing Information Protocol (RIP), and VRRP. Ping and SSH to the concerned interface IP address does not work.

Note The IP Event Dampening feature has no effect on any routing protocols if it is not enabled or an interface is not dampened.

How to Configure IP Event Dampening

Enabling IP Event Dampening

The **dampening** command is entered in interface configuration mode to enable the IP Event Dampening feature. If this command is applied to an interface that already has dampening configured, all dampening states are reset and the accumulated penalty will be set to 0. If the interface has been dampened, the accumulated penalty will fall into the reuse threshold range, and the dampened interface will be made available to the network. The flap counts, however, are retained.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type number*
3. **dampening** [*half-life-period reuse-threshold*] [*suppress-threshold max-suppress* [*restart-penalty*]]
4. **no dampening**
5. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**interface** *type number*	Enters interface configuration mode and configures the specified interface.
Step 3	**dampening** [*half-life-period reuse-threshold*] [*suppress-threshold max-suppress* [*restart-penalty*]]	Enables interface dampening. • Entering the **dampening** command without any arguments enables interface dampening with default configuration parameters. • When manually configuring the timer for the *restart-penalty* argument, the values must be manually entered for all arguments.
Step 4	**no dampening**	Disables interface dampening.
Step 5	**end**	Exits interface configuration mode.

Verifying IP Event Dampening

Use the **show dampening interface** or **show interface dampening** commands to verify the configuration of the IP Event Dampening feature.

SUMMARY STEPS

1. **show ip interface** [**interface**]
2. **show dampening interface**
3. **show interface dampening**

DETAILED STEPS

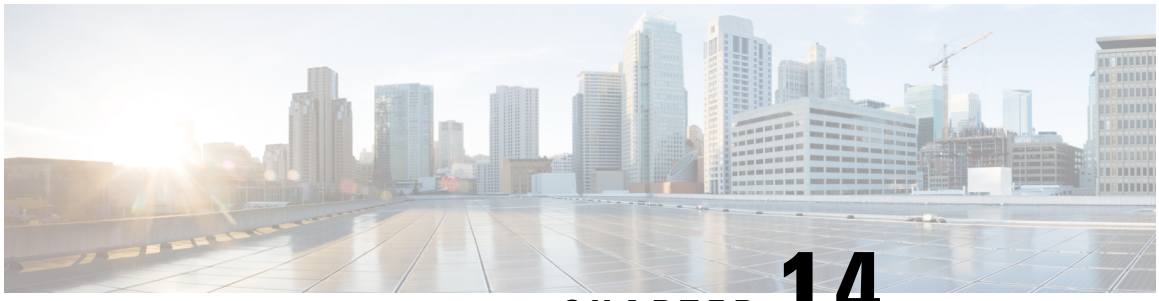
Procedure

	Command or Action	Purpose
Step 1	**show ip interface** [**interface**]	Displays all the configured dampening parameters including penalty information. You see the output only if you have the IP enabled on the interface.
Step 2	**show dampening interface**	Displays dampened interfaces.
Step 3	**show interface dampening**	Displays dampened interfaces on the local router.

Default Settings for IP Dampening Parameters

Table 20: Default values for IP Dampening Parameters

Parameters	Range	Default
Half-life	1-30	5
Reuse threshold	1-20000	800
Suppress threshold	1-20000	2000
Max suppress time	1-255 seconds	20 seconds
Apply restart penalty		False
Restart penalty	true / false	false



CHAPTER 14

Configuring IP TCP MSS

Information About IP TCP MSS

The IP TCP Maximum Segment Size (MSS) feature enables a switch to set a maximum segment size for all TCP connections that originate or terminate at a Cisco Nexus 9000 Series switch. The MSS in a TCP header field is the maximum data size or payload that a host can send or receive in a single segment. By default, a Cisco Nexus 9000 Series switch sets the MSS value to 536 bytes for IPv4 TCP connections and 1240 bytes for IPv6 TCP connections. This default value is set by the switch during the initial TCP connection establishment.

The switch from which the TCP connection originates will always set the MSS to the user-configured MSS or the difference between the route interface MTU and the protocol header, whichever is lower. Thus, Host A sends a SYN packet with the proposed MSS of 1460 bytes to Host B. After receiving the SYN packet with the proposed MSS, Host B sends a SYN-ACK packet to Host A, accepting the proposed MSS value for the TCP connection. Host A sends an ACK packet to Host B, setting the MSS value to 1460 for the TCP connection.

Default Settings for IP TCP MSS

Table 21: Default Settings for IP TCP MSS

Parameter	Default Setting
IP TCP MSS	536 bytes for IPv4 TCP connections 1240 bytes for IPv6 TCP connections

Guidelines and Limitations for IP TCP MSS

If the MSS has to be set to a value that is more than 1460 bytes for IPv4 TCP connections, the corresponding MTU value should be set to the required MSS value plus 40 bytes. If the MSS has to be set to a value that is more than 1440 bytes for IPv6 TCP connections, the corresponding MTU value should be set to the required MSS value plus 60 bytes.

Configuring IP TCP MSS

Setting the MSS for TCP Connections, on page 394

Removing a Set IP TCP MSS, on page 395

Setting the MSS for TCP Connections

Before you begin

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip tcp mss** <*bytes*>
3. switch# **show ip tcp mss**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enter global configuration mode
Step 2	switch(config)# **ip tcp mss** <*bytes*>	Set a maximum segment size.
Step 3	switch# **show ip tcp mss**	Display the configured IP TCP MSS.

Example: Running Configuration

Example

This example shows a running configuration, followed by a verification command that displays the configured IP TCP MSS:

```
configure terminal
 ip tcp mss 5000
 Setting TCP MSS to 5000 bytes

switch# show ip tcp mss
TCP MSS value 5000 bytes
```

Removing a Set IP TCP MSS

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **no ip tcp mss**
3. switch# **show ip tcp mss**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enter global configuration mode
Step 2	switch(config)# **no ip tcp mss**	Remove the configured IP TCP MSS and set the IP TCP MSS to default values.
Step 3	switch# **show ip tcp mss**	Display the configured IP TCP MSS.

Example: Running Configuration

Example

This example shows a running configuration, followed by a verification command that displays the configured IP TCP MSS:

```
configure terminal
 no ip tcp mss 5000
 Setting default MSS value is 536 bytes

switch# show ip tcp mss
TCP MSS value 536 bytes
```

Example: Setting the MSS for TCP Connections

This example shows a setting the MSS for TCP connections:

```
configure terminal
ip tcp mss 2000
```

Example: Removing a Set IP TCP MSS

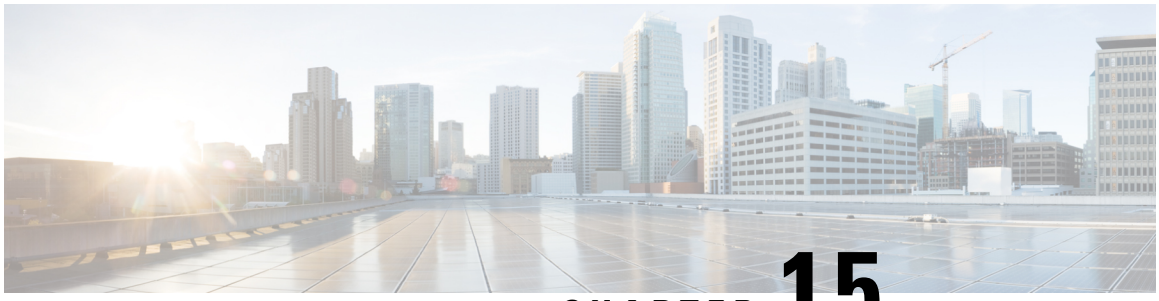
This example shows how to remove the MSS:

```
configure terminal
no ip tcp mss
```

Verifying IP TCP MSS

Table 22: Verifying IP TCP MSS

Command	Purpose
show ip tcp mss	Displays the set IP TCP MSS.



CHAPTER 15

Configuring Unidirectional Ethernet

This chapter describes how to configure Unidirectional Ethernet on the Cisco Nexus 9000 series switches.

Unidirectional Ethernet

Unidirectional Ethernet (UDE) is a network technology that lets you communicate using a single fiber strand for transmitting or receiving data.

With unidirectional links, you can transmit or receive traffic video streaming applications. In these scenarios, most traffic is sent as one-way streams that are not acknowledged.

To create a unidirectional link, configure the port with a bidirectional transceiver so it transmits or receives traffic in one direction.

Use UDE when an appropriate unidirectional transceiver is not available. If transmit-only transceivers are unavailable, configure transmit-only links with software-based UDE.

In certain cases, if you must block all control traffic leaving the interface to prevent a network outage, use the QoS template to block all outgoing traffic on specific Ethernet ports.

Best practices for Unidirectional Ethernet configuration

Use these best practices and recommendations to configure UDE on your Nexus switches

- Configure UDE in send-only mode on your Nexus switches. You *cannot* use UDE receive-only in releases before Cisco NX-OS Release 10.1(2).
- You can enable UDE on all ports at the same time.
- You can use breakout support for UDE starting with Cisco NX-OS Release 10.1(1) and later.
- Port flapping may occur when you configure UDE on a port. You can add physical interfaces with and without UDE configuration into a port-channel. Only add send-only interfaces are added to a port channel.

  If you mix send-only configuration with other interfaces, UDE might *not* work.

- If you configure all members of the port channel as UDE send-only, the port channel may *not* receive packets.
- Special control plane traffic pruning is *not* configurable on send-only ports.
- Unidirectional ports do *not* support features or protocols that require negotiation with the remote port. Disable all features that require bi-directional communication.

Guidelines for UDE Policers

Beginning with Cisco NX-OS Release 10.3(3), you can use QoS template-based UDE. These are the guidelines and limitations for UDE policers.

- Enable the UDE template only on Layer 2 interfaces. Set the port to tap-aggregation mode.
- The policy-map **default-ndb-out-policy** is *not* supported under system QoS. To support this feature, carve the egress Layer 2 QoS TCAM region.

  After you reboot the switch, it might take some time to apply the **default-ndb-out-policy** to the configured interface. During this period, some packets may be forwarded. After the policy is applied, the switch drops all egress control and flood traffic.

  Even if there is no data traffic, the control traffic protocols (such as CDP, LLDP, ARP, and BPDU from the CPU) match the ACL entry and are dropped, which increments the violated count This behavior is expected when when you configure **default-ndb-out-policy**.
- You can use QoS template-based UDE on Cisco Nexus 9300-EX, FX, FX2, FX3, GX, GX2 Series switches, and Cisco Nexus 9500 Series switches with 9700-FX or GX line cards.
- You *cannot* use QoS template on port channels.

UDE support on Nexus switches

- UDE support is available only for native 10G-LR/10G-LRS transceivers. UDE *cannot* be used with QSAs or breakout cables.
- Beginning with Cisco NX-OS Release 10.1(2), UDE is supported on these Cisco Nexus switches:
  - N9K-X9624D-R2
  - N9K-X9636Q-R
  - N9K-X9636C-RX
  - N9K-X96136YC-R
  - N9K-X9624D-R2
  - N9K-X9636C-R
  - Cisco Nexus 3636C-R and Cisco Nexus 36180YC-R modules.
- You can use UDE at the hardware level only on Cisco Nexus 9500 switches with X97160YC-EX line cards
- Beginning with Cisco NX-OS Release 10.1(1), UDE is supported on these switches:
  - Cisco Nexus 9000 EX, FX, FX2 and FX3 platform switches

- N9K-C9336C-FX2
- N9KC93240YC-FX2
- N9K-C93180YC-FX
- N9K-C93360YC-FX2 TOR switches
- N9K-X97160YC-EX line card.

- Beginning with Cisco NX-OS Release 10.1(1), UDE supports the following transceivers: 10G-SR, 10G-AOC, 40G-SR, 40G-LR, 40G-AOC, 100G-SR, 100G-LR, and 100G-AOC.

Configure Unidirectional Ethernet

Configure the ethernet interface for unidirectional communication on the switch. Set the interface to send-only or receive-only mode.

Procedure

	Command or Action	Purpose
Step 1	Enter interface configuration mode using the **interface ethernet** {**type slot /port**} command. **Example:** `switch(config)# interface ethernet 3/1`	
Step 2	Configure send-only mode using the **unidirectional send-only** command. **Example:** `switch(config-if)# unidirectional send-only`	
Step 3	Configure receive-only mode using the **unidirectional receive-only** command. **Example:** `switch(config-if)# unidirectional receive-only`	
Step 4	Exit interface mode using the **exit** command. **Example:** `switch(config)# exit`	
Step 5	Display the running configuration for the interface using the **show running-config interface** {**type slot /port**} command. **Example:** `switch(config)# show running-config interface ethernet 3/1`	

	Command or Action	Purpose
Step 6	Save the configuration using the **copy running-config startup-config** command. **Example:** `switch(config)# copy running-config startup-config`	

You have configured the Ethernet interface for unidirectional operation.

Example

This example shows how to configure an Ethernet interface for send-only unidirectional communication.

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# unidirectional send-only
switch(config-if)# exit
switch(config)# exit
switch#
```

This example shows how to display the running configuration for the interface to verify the unidirectional setting and save the configuration.

```
switch# show running-config interface ethernet 3/1
!
interface ethernet 3/1
  unidirectional send-only
!
```

Configure UDE policers (task)

Block or limit all egress traffic on the Ethernet ports using a Unidirectional Ethernet (UDE) QoS policy.

To configure Unidirectional Ethernet with a QoS template, use these steps.

Procedure

	Command or Action	Purpose
Step 1	Configure the TCAM (Ternary Content Addressable Memory) region for egress Layer 2 QoS to allocate resources using the **hardware access-list tcam region egr-l2-qos 256** command.	Set the size of this region to 256 entries.
Step 2	Save the running configuration (including the TCAM region change) using **copy run start** command.	Saving the changes keeps the configuration after a reload.
Step 3	Reload the switch with the **reload** command to apply the changes for the new TCAM configuration. **Example:**	You must reboot the switch after modifying TCAM regions for the changes to take effect.

	Command or Action	Purpose
	`switch(config)# hardware access-list tcam region egr-l2-qos 256`	

Step 4	Enter interface configuration mode for the Ethernet interface using the **interface type slot/port** command.
	Example:
	`switch(config)# interface Ethernet 1/6`
	`switch(config-if)#`	

Step 5	Apply the UDE QoS service policy to the interface using the **service-policy type qos output default-ndb-out-policy** command.	The switch polices all egress traffic on the Ethernet interface. The switch forwards only traffic that meets the configured parameters and drops traffic that violates them.

The attached QoS policy limits or blocks all egress traffic on the Ethernet port. Only traffic that conforms to the configured policing parameters is forwarded; all traffic that violates these parameters is dropped

What to do next

Verify policy status using **show policy-map type qos default-ndb-out-policy** command.

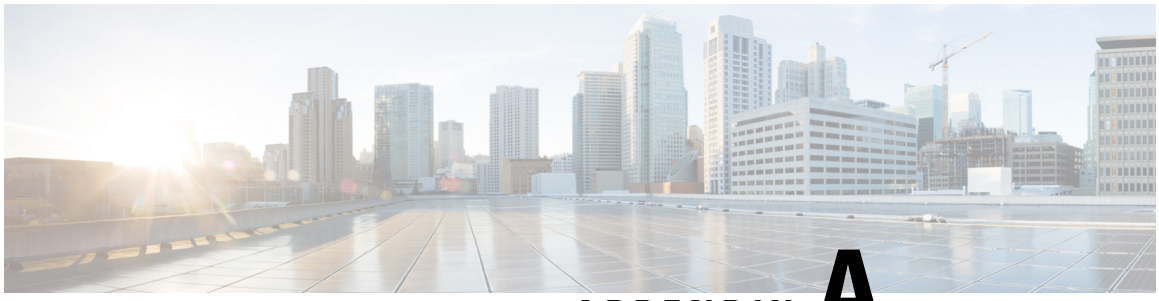
```
switch# show policy-map type qos default-ndb-out-policy

Type qos policy-maps
====================
policy-map type qos default-ndb-out-policy
class class-ndb-default
police cir 0 bps conform transmit violate drop
```

Verify the UDE police statistics for a specific interface.

```
switch# show policy-map interface Ethernet 1/6 output type qos

Global statistics status : enabled
Ethernet1/6
Service-policy (qos) output: default-ndb-out-policy
SNMP Policy Index: 285213501
Class-map (qos): class-ndb-default (match-any)
Slot 1
61211339 packets 15669992128 bytes
5 minute offered rate 17721223780 bps
Aggregate forwarded :
61211339 packets 110848 bytes
police cir 0 bps
conformed 0 bytes, n/a bps action: transmit
violated 15669881280 bytes, n/a bps action: drop
```



APPENDIX A

Configuring Layer 2 Data Center Interconnect

This section contains an example of how to configure a Layer 2 Data Center Interconnect (DCI) with the use of a Virtual Port-Channel (vPC).

Data Center Interconnect (concept)

Data Center Interconnect (DCI) is a set of networking technologies and methodologies that

- link two or more distinct data center facilities over any distance,
- extend specific VLANs and provide Layer 2 adjacency for servers and Network Attached Storage (NAS) devices.

Cisco Nexus 9000 series switches support DCI with FHRP isolation. However DCI with FHRP isolation is not supported on Cisco Nexus 9500 switches with N9K-X9636C-R and N9K-X9636Q-R line cards. Creating a single logical link between multiple sites with vPC allows you to take advantage of the benefits of STP isolation using BPDU filtering across the DCI vPC port-channel. With this configuration, Bridge Protocol Data Unit (BPDU) does not cross between data centers, effectively isolating the STP fault domain between sites.

Note vPC is to interconnect a maximum of two data centers.

DCI Support on Nexus switches

Note The supported platforms include Cisco Nexus 9500 Series switches with N9K-X9636C-R, N9K-X9636Q-R, N9K-X9636C-RX line cards.

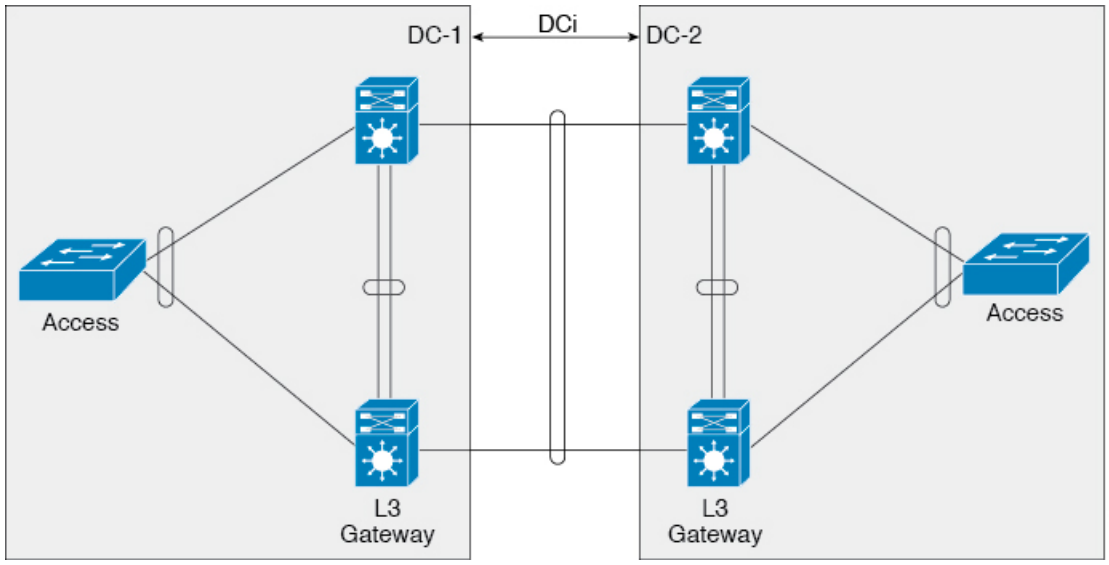
Example of Layer 2 Data Center Interconnect

The following is an example configuration of a Layer 2 Data Center Interconnect (DCI) with use of vPC. The example allows for First Hop Redundancy Protocol (FHRP) isolation.

Note vPC and Hot Standby Routing Protocol (HSRP) have already been configured.

Note Link Aggregation Control Protocol (LACP) should be used on the vPC link, which acts as the DCI.

Figure 38: Dual Layer 2/Layer 3 POD Interconnect

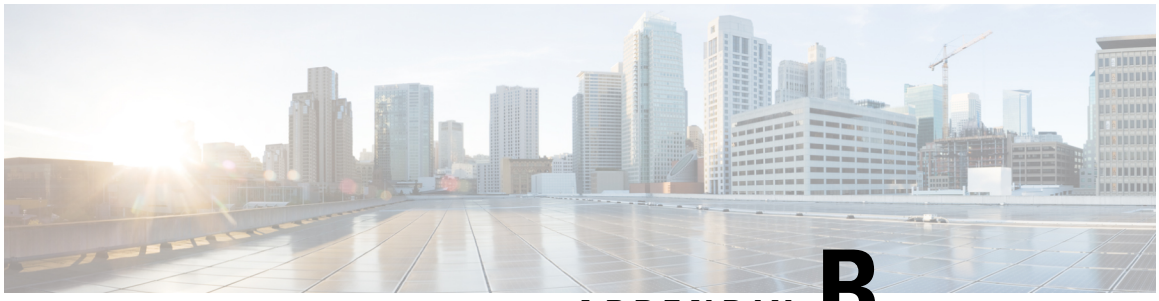


In this example, the Layer 3 (L3) gateway is configured on the same vPC pair and acts as the DCI. In order to isolate the Hot Standby Routing Protocol (HSRP), you must configure a Port Access Control List (PACL) on the DCI port-channel and disable HSRP Gratuitous Address Resolution Protocols (ARPs) (GARPs) on the Switched Virtual Interfaces (SVIs) for the VLANs that move across the DCI.

```
ip access-list DENY_HSRP_IP
  10 deny udp any 224.0.0.2/32 eq 1985
  20 deny udp any 224.0.0.102/32 eq 1985
  30 permit ip any any

interface <DCI-Port-Channel>
  ip port access-group DENY_HSRP_IP in

interface Vlan <x>
  no ip arp gratuitous hsrp duplicate
```



APPENDIX B

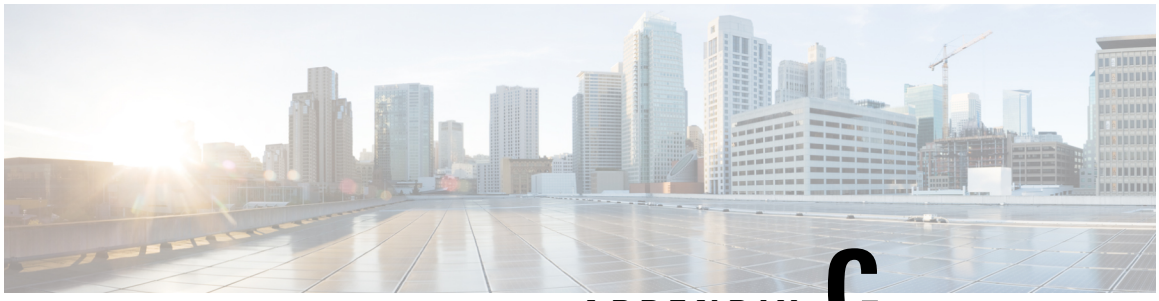
IETF RFCs supported by Cisco NX-OS Interfaces

This appendix lists the IETF RFCs for interfaces supported by Cisco NX-OS.

IPv6 RFCs

RFCs	Title
RFC 2373	*IP Version 6 Addressing Architecture*
RFC 2374	*An Aggregatable Global Unicast Address Format*
RFC 2460	*Internet Protocol, Version 6 (IPv6) Specification*
RFC 2462	*IPv6 Stateless Address Autoconfiguration*
RFC 2464	*Transmission of IPv6 Packets over Ethernet Networks*
RFC 2467	*Transmission of IPv6 Packets over FDDI Networks*
RFC 2472	*IP Version 6 over PPP*
RFC 2492	*IPv6 over ATM Networks*
RFC 2590	*Transmission of IPv6 Packets over Frame Relay Networks Specification*
RFC 3021	*Using 31-Bit Prefixes on IPv4 Point-to-Point Links*
RFC 3152	*Delegation of IP6.ARPA*
RFC 3162	*RADIUS and IPv6*
RFC 3513	*Internet Protocol Version 6 (IPv6) Addressing Architecture*
RFC 3596	*DNS Extensions to Support IP version 6*
RFC 4193	*Unique Local IPv6 Unicast Addresses*

IETF RFCs supported by Cisco NX-OS Interfaces



APPENDIX C

Configuration Limits for Cisco NX-OS Interfaces

The configuration limits are documented in the *Cisco Nexus 9000 Series NX-OS Verified Scalability Guide*.

Configuration Limits for Cisco NX-OS Interfaces

INDEX

I

L

M

N

P

R

S

T

U

V