Cisco Nexus 3000 Series NX-OS Interfaces Configuration Guide, Release 7.x

First Published: 2015-08-10

Last Modified: 2018-02-12

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

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

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

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

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

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

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: http://www.cisco.com/go/trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

© 2018 Cisco Systems, Inc. All rights reserved.

CONTENTS

PREFACE

Preface xiii

Audience xiii

Document Conventions xiii

Related Documentation for Cisco Nexus 3000 Series Switches xiv

Documentation Feedback xiv

Communications, Services, and Additional Information xiv

CHAPTER 1

New and Changed Information 1

New and Changed Information in this Release 1

CHAPTER 2

Configuring Layer 2 Interfaces 5

Licensing Requirements 5

Information About Ethernet Interfaces 5

Interface Command 5

Unidirectional Link Detection Parameter 6

Default UDLD Configuration 7

UDLD Aggressive and Nonaggressive Modes 7

Interface Speed 7

40-Gigabit Ethernet Interface Speed 8

Port Modes 9

SVI Autostate 12

Cisco Discovery Protocol 13

Default CDP Configuration 13

Error-Disabled State 14

Default Interfaces 14

Debounce Timer Parameters 14

MTU Configuration 15
Counter Values 15
Downlink Delay 16
Default Physical Ethernet Settings 16
Configuring Ethernet Interfaces 17
Guidelines for Configuring Ethernet Interfaces 17
Configuring the UDLD Mode 17
Triggering the Link State Consistency Checker 18
Changing an Interface Port Mode 19
Configuring the Interface Speed 21
Configuring Break-Out 10-Gigabit Interface Speed Ports 22
Configuring Break-In 40-Gigabit Ethernet Interface Speed Ports 23
Switching Between QSFP and SFP+ Ports 23
Disabling Link Negotiation 25
Disabling SVI Autostate 26
Configuring a Default Interface 27
Configuring the CDP Characteristics 28
Enabling or Disabling CDP 28
Enabling the Error-Disabled Detection 29
Enabling the Error-Disabled Recovery 30
Configuring the Error-Disabled Recovery Interval 31
Disabling the Error-Disabled Recovery 31
Configuring the Debounce Timer 32
Configuring the Description Parameter 33
Disabling and Restarting Ethernet Interfaces 33
Configuring Downlink Delay 34
Displaying Interface Information 35
MIBs for Layer 2 Interfaces 37

CHAPTER 3 **Configuring Layer 3 Interfaces 39**
Information About Layer 3 Interfaces 39
Routed Interfaces 39
Subinterfaces 40
VLAN Interfaces 41

Changing VRF Membership for an Interface 42
Notes About Changing VRF Membership for an Interface 42
Loopback Interfaces 43
IP Unnumbered 43
Tunnel Interfaces 43
Guidelines and Limitations for Layer 3 Interfaces 43
Default Settings for Layer 3 Interfaces 44
SVI Autostate Disable 44
DHCP Client Discovery 44
Limitations for Using DHCP Client Discovery on Interfaces 45
MAC-Embedded IPv6 Address 45
Configuring Layer 3 Interfaces 45
Configuring a Routed Interface 45
Configuring a Subinterface 46
Configuring the Bandwidth on an Interface 47
Configuring a VLAN Interface 48
Enabling Layer 3 Retention During VRF Membership Change 49
Configuring a Loopback Interface 49
Configuring IP Unnumbered on an Ethernet Interface 50
Configuring OSPF for an IP Unnumbered Interface 51
Configuring ISIS for an IP Unnumbered Interface 52
Assigning an Interface to a VRF 54
Configuring an Interface MAC Address 55
Configuring a MAC-Embedded IPv6 Address 56
Configuring SVI Autostate Disable 58
Configuring a DHCP Client on an Interface 59
Verifying the Layer 3 Interfaces Configuration 59
Triggering the Layer 3 Interface Consistency Checker 60
Monitoring Layer 3 Interfaces 61
Configuration Examples for Layer 3 Interfaces 62
Example of Changing VRF Membership for an Interface 63
Related Documents for Layer 3 Interfaces 65
MIBs for Layer 3 Interfaces 65
Standards for Layer 3 Interfaces 65

Feature History for Layer 3 Interfaces 65

CHAPTER 4 **Configuring Port Channels 67**

Information About Port Channels 67

Understanding Port Channels 68

Compatibility Requirements 68

Load Balancing Using Port Channels 70

Resilient Hashing 72

Hashing for NVGRE Traffic 72

Symmetric Hashing 72

Understanding LACP 73

LACP Overview 73

LACP ID Parameters 74

Channel Modes 74

LACP Marker Responders 75

LACP-Enabled and Static Port Channel Differences 75

LACP Port Channel Minimum Links and MaxBundle 76

Configuring Port Channels 76

Creating a Port Channel 76

Adding a Port to a Port Channel 77

Configuring Load Balancing Using Port Channels 78

Enabling LACP 80

Configuring the Channel Mode for a Port 80

Configuring LACP Port Channel MinLinks 81

Configuring the LACP Port-Channel MaxBundle 82

Configuring the LACP Fast Timer Rate 84

Configuring the LACP System Priority and System ID 84

Configuring the LACP Port Priority 85

Verifying Port Channel Configuration 86

Triggering the Port Channel Membership Consistency Checker 86

Verifying the Load-Balancing Outgoing Port ID 87

Feature History for Port Channels 88

Port Profiles 88

Configuring Port Profiles 89

Creating a Port Profile 89
Entering Port-Profile Configuration Mode and Modifying a Port Profile 90
Assigning a Port Profile to a Range of Interfaces 91
Enabling a Specific Port Profile 92
Inheriting a Port Profile 93
Removing a Port Profile from a Range of Interfaces 93
Removing an Inherited Port Profile 94

CHAPTER 5 **Configuring IP Tunnels** 97
Information About IP Tunnels 97
GRE Tunnels 98
Point-to-Point IP-in-IP Tunnel Encapsulation and Decapsulation 98
Multi-Point IP-in-IP Tunnel Decapsulation 98
Prerequisites for IP Tunnels 98
Guidelines and Limitations for IP Tunnels 98
Default Settings for IP Tunneling 102
Configuring IP Tunnels 102
Enabling Tunneling 102
Creating a Tunnel Interface 103
Configuring a Tunnel Interface 104
Configuring a Tunnel Interface Based on Policy Based Routing 106
Configuring a GRE Tunnel 107
Assigning VRF Membership to a Tunnel Interface 110
Verifying the IP Tunnel Configuration 110
Configuration Examples for IP Tunneling 111
Related Documents for IP Tunnels 111
Standards for IP Tunnels 111
Feature History for Configuring IP Tunnels 112

CHAPTER 6 **Configuring VXLANs** 113
Overview 113
VXLAN Overview 113
VXLAN Encapsulation and Packet Format 114
VXLAN Tunnel Endpoints 114

VXLAN Packet Forwarding Flow 115
VXLAN Implementation on Cisco Nexus 3100 Platform Switches 115
Layer 2 Mechanisms for Broadcast, Unknown Unicast, and Multicast Traffic 115
Layer 2 Mechanisms for Unicast-Learned Traffic 115
VXLAN Layer 2 Gateway as a Transit Multicast Router 116
ECMP and LACP Load Sharing with VXLANs 116
Guidelines and Limitations for VXLANs 116
FHRP Over VXLAN 118
Overview of FHRP over VXLAN 118
Guidelines and Limitations for FHRP Over VXLAN 118
FHRP Over VXLAN Topology 119
Considerations for VXLAN Deployment 120
vPC Guidelines and Limitations for VXLAN Deployment 120
Configuring VXLAN Traffic Forwarding 122
Enabling and Configuring the PIM Feature 123
Configuring a Rendezvous Point 123
Enabling a VXLAN 124
Mapping a VLAN to a VXLAN VNI 125
Configuring a Routing Protocol for NVE Unicast Addresses 125
Creating a VXLAN Destination UDP Port 126
Creating and Configuring an NVE Interface 127
Configuring Replication for a VNI 128
Configuring Multicast Replication 128
Configuring Ingress Replication 128
Configuring Q-in-VNI 129
Verifying the VXLAN Configuration 131
Overview of IGMP Snooping Over VXLAN 133
Guidelines and Limitations for IGMP Snooping Over VXLAN 133
Configuring IGMP Snooping Over VXLAN 133

CHAPTER 7 **Configuring VXLAN BGP EVPN 135**
Information About VXLAN BGP EVPN 135
Guidelines and Limitations for VXLAN BGP EVPN 135
Notes for EVPN Convergence 137

Considerations for VXLAN BGP EVPN Deployment 138
vPC Considerations for VXLAN BGP EVPN Deployment 139
Network Considerations for VXLAN Deployments 141
Considerations for the Transport Network 142
Considerations for Tunneling VXLAN 143
BGP EVPN Considerations for VXLAN Deployment 143
Configuring VXLAN BGP EVPN 145
Enabling VXLAN 145
Configuring VLAN and VXLAN VNI 146
Configuring VRF for VXLAN Routing 146
About RD Auto 147
About Route-Target Auto 147
Configuring SVI for Hosts for VXLAN Routing 148
Configuring VRF Overlay VLAN for VXLAN Routing 148
Configuring VNI Under VRF for VXLAN Routing 148
Configuring Anycast Gateway for VXLAN Routing 149
Configuring the NVE Interface and VNIs 149
Configuring BGP on the VTEP 149
Configuring RD and Route Targets for VXLAN Bridging 150
About RD Auto 151
About Route-Target Auto 151
Configuring BGP for EVPN on the Spine 152
Suppressing ARP 153
Disabling VXLANs 154
Duplicate Detection for IP and MAC Addresses 154
Enabling Nuage Controller Interoperability 155
Verifying the VXLAN BGP EVPN Configuration 156
Example of VXLAN BGP EVPN (EBGP) 157
Example of VXLAN BGP EVPN (IBGP) 166
Example Show Commands 175

CHAPTER 8 **Configuring VXLAN OAM 179**
VXLAN OAM Overview 179
Loopback (Ping) Message 180

Traceroute or Pathtrace Message 181
Configuring VXLAN OAM 183
Configuring NGOAM Profile 186
NGOAM Authentication 187

CHAPTER 9 **Configuring VXLAN Multihoming** 189
VXLAN EVPN Multihoming Overview 189
Introduction to Multihoming 189
BGP EVPN Multihoming Terminology 189
EVPN Multihoming Implementation 190
EVPN Multihoming Redundancy Group 191
Ethernet Segment Identifier 191
LACP Bundling 191
Guidelines and Limitations for VXLAN EVPN Multihoming 192
Configuring VXLAN EVPN Multihoming 192
Enabling EVPN Multihoming 192
VXLAN EVPN Multihoming Configuration Examples 193
Configuring Layer 2 Gateway STP 195
Layer 2 Gateway STP Overview 195
Guidelines for Moving to Layer 2 Gateway STP 195
Enabling Layer 2 Gateway STP on a Switch 196
Configuring VXLAN EVPN Multihoming Traffic Flows 199
EVPN Multihoming Local Traffic Flows 199
EVPN Multihoming Remote Traffic Flows 203
EVPN Multihoming BUM Flows 208
Configuring VLAN Consistency Checking 211
Overview of VLAN Consistency Checking 211
VLAN Consistency Checking Guidelines and Limitations 212
Configuring VLAN Consistency Checking 212
Displaying Show command Output for VLAN Consistency Checking 212
Configuring ESI ARP Suppression 214
Overview of ESI ARP Suppression 214
Limitations for ESI ARP Suppression 214
Configuring ESI ARP Suppression 214

Displaying Show Commands for ESI ARP Suppression 215

CHAPTER 10 IPv6 Across a VXLAN EVPN Fabric 217

Overview of IPv6 Across a VXLAN EVPN Fabric 217

Configuring IPv6 Across a VXLAN EVPN Fabric Example 217

Show Command Examples 220

CHAPTER 11 Configuring Virtual Port Channels 223

Information About vPCs 223

vPC Overview 223

Terminology 224

vPC Terminology 224

vPC Domain 225

Peer-Keepalive Link and Messages 225

Compatibility Parameters for vPC Peer Links 226

Configuration Parameters That Must Be Identical 227

Configuration Parameters That Should Be Identical 228

Per-VLAN Consistency Check 228

vPC Auto-Recovery 228

vPC Peer Links 228

vPC Peer Link Overview 229

vPC Number 230

vPC Interactions with Other Features 230

vPC and LACP 230

vPC Peer Links and STP 230

CFSoE 231

Guidelines and Limitations for vPCs 231

Verifying the vPC Configuration 232

Viewing the Graceful Type-1 Check Status 233

Viewing a Global Type-1 Inconsistency 234

Viewing an Interface-Specific Type-1 Inconsistency 235

Viewing a Per-VLAN Consistency Status 236

vPC Default Settings 238

Configuring vPCs 239

Enabling vPCs 239
Disabling vPCs 239
Creating a vPC Domain 240
Configuring a vPC Keepalive Link and Messages 241
Creating a vPC Peer Link 243
Checking the Configuration Compatibility 244
Enabling vPC Auto-Recovery 245
Configuring the Restore Time Delay 246
Configuring Delay Restore on an Orphan Port 246
Configuring the Suspension of Orphan Ports 247
Excluding VLAN Interfaces from Shutting Down a vPC Peer Link Fails 248
Configuring the VRF Name 249
Moving Other Port Channels into a vPC 250
Manually Configuring a vPC Domain MAC Address 251
Manually Configuring the System Priority 251
Manually Configuring a vPC Peer Switch Role 252
Configuring Layer 3 over vPC 253

CHAPTER 12 **Configuring Q-in-Q VLAN Tunnels** 255
Information About Q-in-Q Tunnels 255
Native VLAN Hazard 257
Information About Layer 2 Protocol Tunneling 258
Guidelines and Limitations for Q-in-Q Tunneling 260
Configuring Q-in-Q Tunnels and Layer 2 Protocol Tunneling 261
Creating a 802.1Q Tunnel Port 261
Enabling the Layer 2 Protocol Tunnel 262
Configuring Thresholds for Layer 2 Protocol Tunnel Ports 262
Configuring VLAN Mapping for Selective Q-in-Q on a 802.1Q Tunnel Port 264
Verifying the Q-in-Q Configuration 265
Configuration Example for Q-in-Q and Layer 2 Protocol Tunneling 265
Feature History for Q-in-Q Tunnels and Layer 2 Protocol Tunneling 266

Preface

This preface includes the following sections:

- Audience, on page xiii
- Document Conventions, on page xiii
- Related Documentation for Cisco Nexus 3000 Series Switches, on page xiv
- Documentation Feedback, on page xiv
- Communications, Services, and Additional Information, on page xiv

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 the user supplies the values.
[x]	Square brackets enclose an optional element (keyword or argument).
[x | y]	Square brackets enclosing keywords or arguments separated by a vertical bar indicate an optional choice.
{x | y}	Braces enclosing keywords or arguments separated by a vertical bar indicate a required choice.
[x {y | z}]	Nested set of square brackets or braces indicate optional or required choices within optional or required elements. Braces and a vertical bar within square brackets indicate a required choice within an optional element.

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 will include the quotation marks.

Examples use the following conventions:

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

Related Documentation for Cisco Nexus 3000 Series Switches

The entire Cisco Nexus 3000 Series switch documentation set is available at the following URL:

https://www.cisco.com/c/en/us/support/switches/nexus-3000-series-switches/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 nexus3k-docfeedback@cisco.com. We appreciate your feedback.

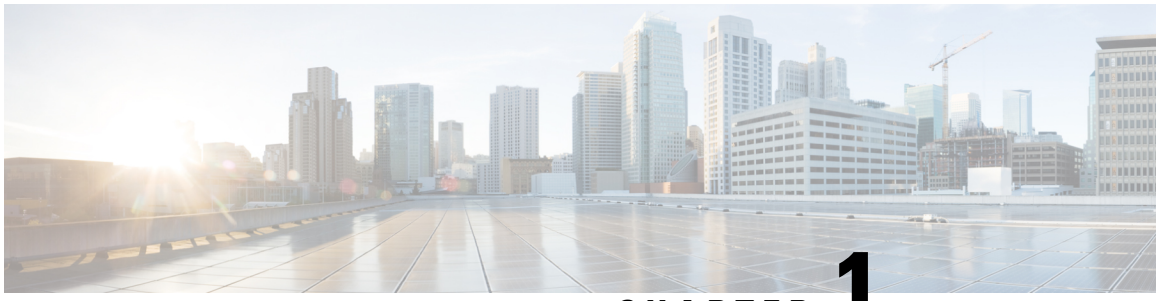
Communications, Services, and Additional Information

- 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 Marketplace.
- 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 web-based tool that acts as a gateway to the Cisco bug tracking system that maintains a comprehensive list of defects and vulnerabilities in Cisco products and software. BST provides you with detailed defect information about your products and software.

Preface



CHAPTER 1

New and Changed Information

This chapter contains the following sections:

- New and Changed Information in this Release, on page 1

New and Changed Information in this Release

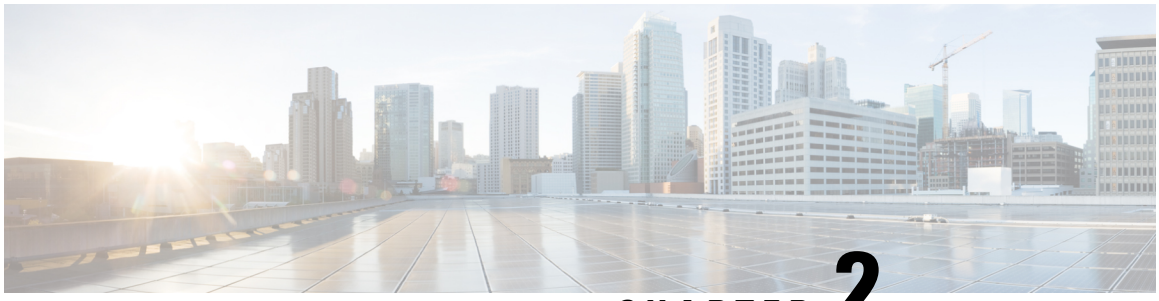
The following table provides an overview of the significant changes made to this configuration guide. The table does not provide an exhaustive list of all the changes made to this guide or all new features in a particular release.

Feature	Description	Added or Changed in Release	Where Documented
Configuring Delay Restore on Orphan Ports	Added support to configure Delay Restore on orphan Ports	7.0(3)I7(1)	Configuring Delay Restore on an Orphan Port, on page 246
Configuring the Suspension of Orphan Ports	Added support to configure suspension of orphan Ports	7.0(3)I7(1)	Configuring the Suspension of Orphan Ports, on page 247
Support for *speed-auto*	Added support for *speed-auto* as default port speed.	7.0(3)I7(1)	40-Gigabit Ethernet Interface Speed, on page 8
Configuring FHRP Over VXLAN	Added support for configuring FHRP	7.0(3)I7(1)	FHRP Over VXLAN, on page 118
HSRP over VXLAN	Added Support for HSRP over VXLAN	7.0(3)I7(1)	Guidelines and Limitations for VXLANs, on page 116
VXLAN EVPN Multihoming	Added support for VXLAN EVPN Multihoming on Cisco Nexus 3100 Series Switches	7.0(3)I7(1)	Configuring VXLAN Multihoming, on page 189

Feature	Description	Added or Changed in Release	Where Documented
Configuring IP-in-IP tunnel decapsulation on directly connected IP addresses	Added support for configuring IP-in-IP tunnel decapsulation on any directly connected IP addresses.	7.0(3)I6(1)	Guidelines and Limitations for IP Tunnels, on page 98and Configuring a Tunnel Interface, on page 104
VXLAN Support	Added VXLAN support on Cisco Nexus 3232C and 3264Q switches.	7.0(3)I6(1)	Guidelines and Limitations for VXLANs, on page 116
IP unnumbered feature support on port channels.	Added support for IP unnumbered on port channel interfaces	7.0(3)I6(1)	Information About Layer 3 Interfaces, on page 39
Layer 3 over vPC	Added support for Layer 3 over vPC	7.0(3)I5(1)	Configuring Layer 3 over vPC, on page 253
Support for breakout port modes.	Introduced breakout port modes for the Cisco Nexus 3132Q-V, 31108PC-V, and 31108TC-V switches.	7.0(3)I4(2)	Port Modes
Support for changing VRF membership for an SVI	Added support for changing VRF membership for an SVI.	7.0(3)I4(1)	Changing VRF Membership for an Interface, on page 42
Configuring port profiles	Added support for port profiles.	7.0(3)I4(1)	Port Profiles, on page 88
Configuring VXLAN BGP EVPN	Added support for configuring VXLAN BGP EVPN.	7.0(3)I4(1)	BGP EVPN Considerations for VXLAN Deployment, on page 143
IPv6 Across a VXLAN EVPN Fabric	Added support for IPv6 Across a VXLAN EVPN Fabric	7.0(3)I4(1)	Overview of IPv6 Across a VXLAN EVPN Fabric, on page 217
IP unnumbered.	Added support for IP unnumbered command.	7.0(3)I3(1)	IP Unnumbered, on page 43 Configuring IP Unnumbered on an Ethernet Interface, on page 50
Guidelines for configuring ethernet interfaces.	Added guidelines for configuring ethernet interfaces.	7.0(3)I2(1)	Guidelines for Configuring Ethernet Interfaces, on page 17

Feature	Description	Added or Changed in Release	Where Documented
Duplicate ports on VXLAN multicast encapsulation path.	VXLAN multicast encapsulation path has duplicate ports after reloading the VPC peers.	7.0(3)I2(1)	vPC Guidelines and Limitations for VXLAN Deployment, on page 120
Configuring the interface breakout when changing the portmode from QSFP to SFP+.	Added information on configuring the interface breakout when changing the portmode from QSFP to SFP+	7.0(3)I2(1)	Switching Between QSFP and SFP+ Ports, on page 23
Updated the running-config output for configuring breakout on QSFP ports.	Configuring breakout on QSFP ports using **speed 10000** adds **interface breakout module module number port port range map 10g-4x** in the running-config output.	7.0(3)I2(1)	Configuring Break-Out 10-Gigabit Interface Speed Ports, on page 22
Updated output of the **sh vpc brief** CLI command.	The output of the **sh vpc brief** CLI command displays two additional fields, Delay-restore status and Delay-restore SVI status	7.0(3)I2(1)	Guidelines and Limitations for vPCs, on page 231 Viewing a Per-VLAN Consistency Status, on page 236 Viewing the Graceful Type-1 Check Status, on page 233 Viewing an Interface-Specific Type-1 Inconsistency, on page 235
Support for LACP min-links.	The maximum value of LACP min-links supported is 16.	7.0(3)I2(1)	Configuring LACP Port Channel MinLinks, on page 81

Feature	Description	Added or Changed in Release	Where Documented
Breakout ports are in administratively enabled state after the breakout of the ports into 4x10G mode or the breakin of the ports into 40G mode.	The breakout ports are in administratively enabled state after the breakout of the ports into 4x10G mode or the breakin of the ports into 40G mode. On upgrade from the earlier releases, the configuration restored takes care of restoring the appropriate administrative state of the ports.	7.0(3)I2(1)	40-Gigabit Ethernet Interface Speed, on page 8
The VLAN/SVI is not removed from the Layer 3 interface table, after the configuration is removed.	The VLAN/SVI is not removed from the Layer 3 interface table, after the configuration is removed. The VLAN itself should be removed from the Layer 3 interface table.	7.0(3)I2(1)	Guidelines and Limitations for Layer 3 Interfaces, on page 43
Setting the LACP rate on the ports	You can set the LACP rate only on the ports that are administratively down.	7.0(3)I2(1)	Configuring the LACP Fast Timer Rate, on page 84
VxLAN multicast group scale	It is recommended that the summation of the number of the multicast groups and the OIFLs to be used in a scaled environment should not exclude 1024 which is the current range of the multicast VXLAN VP.	7.0(3)I2(1)	Guidelines and Limitations for VXLANs, on page 116
The regex and source-interface command options.	Added information on regex and source-interface command options.	7.0(3)I2(1)	Guidelines for Configuring Ethernet Interfaces, on page 17



CHAPTER 2

Configuring Layer 2 Interfaces

This chapter contains the following sections:

- Licensing Requirements, on page 5
- Information About Ethernet Interfaces, on page 5
- Default Physical Ethernet Settings , on page 16
- Configuring Ethernet Interfaces, on page 17
- Displaying Interface Information, on page 35
- MIBs for Layer 2 Interfaces, on page 37

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*.

Information About Ethernet Interfaces

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

The Ethernet interfaces are enabled by default.

Interface Command

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

- Interface type—All physical Ethernet interfaces use the **ethernet** keyword.
- Slot number:
  - Slot 1 includes all the fixed ports.
  - Slot 2 includes the ports on the upper expansion module (if populated).
  - Slot 3 includes the ports on the lower expansion module (if populated).
  - Slot 4 includes the ports on the lower expansion module (if populated).

- Port number— Port number within the group.

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

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

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

Unidirectional Link Detection Parameter

The Cisco-proprietary Unidirectional Link Detection (UDLD) protocol allows ports that are connected through fiber optics or copper (for example, Category 5 cabling) Ethernet cables to monitor the physical configuration of the cables and detect when a unidirectional link exists. When the switch detects a unidirectional link, UDLD shuts down the affected LAN port and alerts the user. Unidirectional links can cause a variety of problems, including spanning tree topology loops.

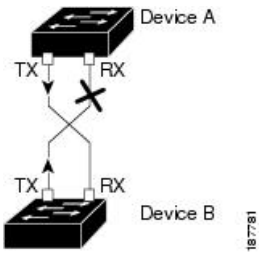
UDLD is a Layer 2 protocol that works with the Layer 1 protocols to determine the physical status of a link. At Layer 1, autonegotiation takes care of physical signaling and fault detection. UDLD performs tasks that autonegotiation cannot perform, such as detecting the identities of neighbors and shutting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

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

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

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

Figure 1: Unidirectional Link



Default UDLD Configuration

The following table shows the default UDLD configuration.

Table 1: UDLD Default Configuration

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

UDLD Aggressive and Nonaggressive Modes

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

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

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

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

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

Interface Speed

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

Note If you set a port configuration that does not use all of the ports, the unused ports are left in the removed state. For example, if you configure 96 x 25G + 32 x 100G on a Cisco Nexus 3264C-E platform switch, the configuration uses 56 ports and leaves 8 ports in the removed state.

```
switch(config)# hardware profile portmode ?
  128x25g         128x25G port mode
  64x100g         64 100G ports with 2x50G, 1x100G, 1x40G capability
  96x25g+32x100g  96x25G+32x100G port mode
```

Where:

- 128x25g: Only 32 QSFP ports are usable
- 64x100g: All 64 ports are usable (default port mode)
- 96x25g+32x100g: Only 56 ports are usable

40-Gigabit Ethernet Interface Speed

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

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

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

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

The ability to break out a 40-Gigabit Ethernet port into four 10-Gigabit Ethernet ports and break in four 10-Gigabit Ethernet ports into a 40-Gigabit Ethernet port dynamically allows you to use any of the

breakout-capable ports to work in the 40-Gigabit Ethernet or 10-Gigabit Ethernet modes without permanently defining them.

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

When using a QSFP-40G-CR4 on Cisco Nexus 3000 switches, you must configure the default speed as 40G in the auto-negotiation parameters. Otherwise, the interface may not be able to bring the link up.

Port Modes

Nexus 3100 Series Switches	Ports	Port Modes
Cisco Nexus 3132Q	32 x QSFP ports and 4 SFP+ ports	The following port modes support breakout: • 32x40G—This is an oversubscribed port mode. All 32 ports are oversubscribed and the first 24 QSFP ports are break-out capable. You cannot enter the **speed 10000** command on ports 25 through 32. 32x40G breakout mode is the default port mode. • 26x40G—This is an oversubscribed port mode. Of the 26 ports, 12 ports are nonoversubscribed (cut-through). These ports are 2,4 to 8,14,and 16 to 20. The remaining 14 ports are oversubscribed. All available QSFP ports are break-out capable. • 24x40G—This is the only nonoversubscribed (cut-through) mode. All available QSFP ports are break-out capable. The Fixed32x40G port mode does not support breakout.

Nexus 3100 Series Switches	Ports	Port Modes
Cisco Nexus 3132Q-V	32 x 40G QSFP ports	• 32x40G—This is the default port mode. Of the 32 ports, first 24 QSFP ports are 10Gx4 break-out capable and the last 8 QSFP ports has a fixed speed of 40G. The maximum port counts are: 96x10G + 8x40G. All ports are oversubscribed equally. The 10G ports does not support cut-through switching. • 26x40G—This port mode supports the maximum number of 10G ports. The first 26 QSFP ports are 10Gx4 break-out capable and the last 6 QSFP ports are not usable. The maximum port counts are: 104x10G. All available QSFP ports are break-out capable. Of the 26 ports, 12 ports are non-oversubscribed (cut-through). These ports are 2,4 to 8,14,and 16 to 20. The remaining 14 ports are oversubscribed. • 24x40G—This is a non-oversubscribed, line rate port mode. Of the 32 ports, first 24 QSFP ports are 10Gx4 break-out capable. The maximum port counts can be 96x10G. The 10G ports support cut-through switching.

Nexus 3100 Series Switches	Ports	Port Modes
Cisco Nexus 31108PC-V	48 x 10G SFP+ ports and 6 x 100G QSFP ports	The following two port modes are supported: • 48x10G SFP+ ports + 6x100G QSFP ports. • 48x10G SFP+ ports + 4x100G QSFP ports + 2x40G QSFP ports. The following features are specific to port mode-1: • SFP+ Ports 1 through 8 always delivers at line-rate. • SFP+ Ports 9 through 48 are always over subscribed. • QSFP Ports 49 through 52 are capable of 100G and/or 40G and 10Gx4 breakout. • QSFP Ports 53 through 54 are capable of 100G and/or 40G. These ports do not support 10Gx4 breakout. • The maximum supported port count is 64x10G + 2x40G/100G. The following features are specific to port mode-2: • SFP+ Ports 1 through 48 always delivers at line-rate and support cut-through switching. • Line-rate SFP+ ports support cut-through switching and because of that it will have less latency. • QSFP Ports 49 through 52 are capable of 100G/40G and 10Gx4 breakout. • QSFP Ports 53 through 54 are capable of 40G and 10Gx4 breakout. These ports do not support 100G. • The maximum supported port count is 72x10G.

Nexus 3100 Series Switches	Ports	Port Modes
Cisco Nexus 31108TC-V	48 x 10GBase-T and 6 x 100G ports	The following two port modes are supported: • 48x10GBASE-T ports + 6x100G QSFP ports. • 48x10GBASE-T ports + 4x100G QSFP ports + 2x40G QSFP ports. The following features are specific to port mode-1: • 10GBASE-T Ports 1 through 8 always delivers at line-rate. • 10GBASE-T Ports 9 through 48 are always over subscribed. • QSFP Ports 49 through 52 are capable of 100G and/or 40G and 10Gx4 breakout. • QSFP Ports 53 through 54 are capable of 100G and/or 40G. These ports do not support 10Gx4 breakout. • The maximum supported port count is 64x10G + 2x40G/100G. The following features are specific to port mode-2: • 10GBASE-T Ports 1 through 48 always delivers at line-rate and support cut-through switching. • Line-rate SFP+ ports support cut-through switching and because of that it will have less latency. • QSFP Ports 49 through 52 are capable of 100G/40G and 10Gx4 breakout. • QSFP Ports 53 through 54 are capable of 40G and 10Gx4 breakout. These ports do not support 100G. • The maximum supported port count is 72x10G.
Cisco Nexus 3172PQ	6 x QSFP ports and 48 SFP+ ports	The following is the default port mode and supports breakout: • 48x10G+breakout6x40G The following are the fixed port modes that do not support breakout: • 48x10G+6x40G • 72x10G

SVI Autostate

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

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

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

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

Cisco Discovery Protocol

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

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

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

The switch supports both CDP Version 1 and Version 2.

Default CDP Configuration

The following table shows the default CDP configuration.

Table 2: Default CDP Configuration

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

Error-Disabled State

An interface is in the error-disabled (err-disabled) state when the inteface is enabled administratively (using the **no shutdown** command) but disabled at runtime by any process. For example, if UDLD detects a unidirectional link, the interface is shut down at runtime. However, because the interface is administratively enabled, the interface status displays as err-disabled. Once an interface goes into the err-disabled state, you must manually reenable it or you can configure an automatic timeout recovery value. The err-disabled detection is enabled by default for all causes. The automatic recovery is not configured by default.

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

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

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

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

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

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

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

Default Interfaces

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

Debounce Timer Parameters

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

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

MTU Configuration

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

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

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

Configuration	Packet Size	Incremented Counters	Traffic
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.

Downlink Delay

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

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

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

Default Physical Ethernet Settings

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

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

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

Configuring Ethernet Interfaces

Guidelines for Configuring Ethernet Interfaces

Configuring the UDLD Mode

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

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

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

Procedure

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

Example

This example shows how to enable UDLD for the switch:

```
switch# configure terminal
```

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

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

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

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

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

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

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

This example shows how to disable UDLD for the switch:

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

Triggering the Link State Consistency Checker

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

Procedure

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

Example

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

```
switch# show consistency-checker link-state module 1
Link State Checks: Link state only
Consistency Check: FAILED
No inconsistencies found for:
   Ethernet1/1
   Ethernet1/2
   Ethernet1/3
   Ethernet1/4
   Ethernet1/5
```

```
   Ethernet1/6
   Ethernet1/7
   Ethernet1/8
   Ethernet1/9
   Ethernet1/10
   Ethernet1/12
   Ethernet1/13
   Ethernet1/14
   Ethernet1/15
Inconsistencies found for following interfaces:
   Ethernet1/11
```

Changing an Interface Port Mode

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

Procedure

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

Example

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

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

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

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

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

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

```
hardware profile tcam region vacl 512
hardware profile portmode 48x10G+4x40G
snmp-server user admin network-admin auth md5 0xdd1d21ee42e93106836cdefd1a60e062
<--Output truncated-->
switch#
```

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

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

Configuring the Interface Speed

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

Procedure

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

Example

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

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

Configuring Break-Out 10-Gigabit Interface Speed Ports

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface** *type slot/port-range*	Enters interface configuration mode for the specified interface. **Note** Interface range is not supported for 40-Gigabit Ethernet interfaces. For example, Eth 1/2-5 is not supported.
Step 3	switch(config-if)# **speed 10000**	Sets the speed on the interface to 10-Gigabit per second. **Note** Configuring breakout on QSFP ports using **speed 10000** adds **interface breakout module <module number> port <port range> map 10g-4x** in the running-config output.

Example

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

```
switch# configure terminal
switch(config)# interface ethernet 1/49
switch(config-if)# speed 10000

switch(config-if)# sh running-config | grep port
  limit-resource port-channel minimum 0 maximum 511
interface breakout module 1 port 49 map 10g-4x ----------------------------------------->
 interface breakout is added on "speed" config
hardware profile portmode 48x10g+breakout6x40g
(config-if)#

(config)# int ethernet 1/49/1
(config-if)#no speed 10000 ----------------------------------> on "no speed", the interface
 breakout cmd is removed.

(config-if)#  sh running-config | grep port
  limit-resource port-channel minimum 0 maximum 511
hardware profile portmode 48x10g+breakout6x40g
```

Configuring Break-In 40-Gigabit Ethernet Interface Speed Ports

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

Procedure

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

Example

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

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

Switching Between QSFP and SFP+ Ports

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

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

Procedure

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

	Command or Action	Purpose
		Note If the first QSFP port speed is 40 Gbps, this command will run, but the SFP+ ports will not become active until after the speed is changed to 10 Gbps.
Step 3	switch(config)# **interface breakout module** *module number* **port** *port range***map 10g-4x**	Enables you to configure the module in 10g mode. When you are changing the portmode from QSFP to SFP+, the **hardware profile front portmode** command takes effect only after breaking out the first QSFP port as displayed in this command.
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 change the portmode from QSFP to SFP+:

```
switch# show int e1/1 tranceiver
Ethernet1/1
transceiver is present
type is QSFP-40G-SR
name is CISCO
part number is AFBR-79EIPZ-CS1
revision is 02
serial number is AVP1645S1QT
nominal bitrate is 10300 MBit/sec per channel
Link length supported for 50/125um fiber is 30 m
Link length supported for 50/125um fiber is 100 m
cisco id is --
cisco extended id number is 16

switch# show running-config | inc portmode
hardware profile portmode 32X40G
hardware profile front portmode qsfp

switch# configure terminal
switch(config)# hardware profile front portmode sfp-plus
switch(config)# interface breakout module 1 port 1 map 10g-4x
switch(config)# copy running-config startup-config
```

This example shows how to make the QSFP port active:

```
switch# configure terminal
switch(config)# hardware profile front portmode qsfp
switch(config)# copy running-config startup-config
```

Disabling Link Negotiation

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

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

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

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

Note 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:

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

Note For 1G SFP with autonegotiation enabled, Auto Speed mode is not compatible to linkup with a peer having 100M speed.

Procedure

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

Example

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

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

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

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

Disabling SVI Autostate

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

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

Note Autostate behavior is enabled by default.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **feature interface-vlan**	Enables the interface-vlan feature.
Step 3	Required: switch(config)# **system default interface-vlan** [**no**] **autostate**	Configures the system to enable or disable the Autostate default behavior.
Step 4	(Optional) switch(config)# **interface vlan** *interface-vlan-number*	Creates a VLAN interface. The number range is from 1 to 4094.
Step 5	(Optional) switch(config-if)# [**no**] **autostate**	Enables or disables Autostate behavior per SVI.
Step 6	(Optional) switch(config)# **show interface-vlan** *interface-vlan*	Displays the enabled or disabled Autostate behavior of the SVI.
Step 7	(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 disable the systems Autostate default for all the SVIs on the switch:

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

This example shows how to enable the systems autostate configuration:

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

Configuring a Default Interface

The default interface feature allows you to clear the existing configuration of multiple interfaces such as Ethernet, loopback, management, VLAN, and port-channel interfaces. All user configuration under a specified interface will be deleted. On a Cisco Nexus C3408-S switch, the number of interfaces you can configure using the **default interface ethernet** command, at a time, is limited to a maximum of 64 ports.

Procedure

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

Example

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

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

Configuring the CDP Characteristics

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	(Optional) switch(config)# [**no**] **cdp advertise** {**v1** | **v2** }	Configures the version to use to send CDP advertisements. Version-2 is the default state. Use the **no** form of the command to return to its default setting.
Step 3	(Optional) switch(config)# [**no**] **cdp format device-id** {**mac-address** | **serial-number** | **system-name**}	Configures the format of the CDP device ID. The default is the system name, which can be expressed as a fully qualified domain name. Use the **no** form of the command to return to its default setting.
Step 4	(Optional) switch(config)# [**no**] **cdp holdtime** *seconds*	Specifies the amount of time a receiving device should hold the information sent by your device before discarding it. The range is 10 to 255 seconds; the default is 180 seconds. Use the **no** form of the command to return to its default setting.
Step 5	(Optional) switch(config)# [**no**] **cdp timer** *seconds*	Sets the transmission frequency of CDP updates in seconds. The range is 5 to 254; the default is 60 seconds. Use the **no** form of the command to return to its default setting.

Example

This example shows how to configure CDP characteristics:

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

Enabling or Disabling CDP

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

Procedure

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

Example

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

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

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

Enabling the Error-Disabled Detection

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

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **errdisable detect cause** {*all* | *link-flap* | *loopback*}	Specifies a condition under which to place the interface in an err-disabled state. The default is enabled.
Step 3	switch(config)# **shutdown**	Brings the interface down administratively. To manually recover the interface from the err-disabled state, enter this command first.

	Command or Action	Purpose
Step 4	switch(config)# **no shutdown**	Brings the interface up administratively and enables the interface to recover manually from the err-disabled state.
Step 5	switch(config)# **show interface status err-disabled**	Displays information about err-disabled interfaces.
Step 6	(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 enable the err-disabled detection in all cases:

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

Enabling the Error-Disabled Recovery

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **errdisable recovery cause** {*all* | *udld* | *bpduguard* | *link-flap* | *failed-port-state* | *pause-rate-limit* | *loopback*}	Specifies a condition under which the interface automatically recovers from the err-disabled state, and the device retries bringing the interface up. The device waits 300 seconds to retry. The default is disabled.
Step 3	switch(config)# **show interface status err-disabled**	Displays information about err-disabled interfaces.
Step 4	(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 enable err-disabled recovery under all conditions:

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

Configuring the Error-Disabled Recovery Interval

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

Procedure

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

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

Disabling the Error-Disabled Recovery

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

Procedure

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

	Command or Action	Purpose
Step 2	switch(config)# **no errdisable recovery cause** {*all* | *udld* | *bpduguard* | *link-flap* | *failed-port-state* | *pause-rate-limit* | *loopback*}	Specifies a condition under which the interface reverts back to the default err-disabled state.
Step 3	(Optional) switch(config)# **show interface status err-disabled**	Displays information about err-disabled interfaces.
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 disable err-disabled recovery:

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

Configuring the Debounce Timer

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

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.

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

Procedure

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

	Command or Action	Purpose
		Disables the debounce timer if you specify 0 milliseconds.

Example

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

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

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

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

Configuring the Description Parameter

You can provide textual interface descriptions for the Ethernet ports.

Procedure

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

Example

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

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

Disabling and Restarting Ethernet Interfaces

You can shut down and restart an Ethernet interface. This action disables all of the interface functions and marks the interface as being down on all monitoring displays.

Procedure

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

Example

This example shows how to disable an Ethernet port:

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

This example shows how to restart an Ethernet interface:

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

Configuring Downlink Delay

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

Procedure

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

Example

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

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

Displaying Interface Information

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

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

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

This example shows how to display the physical Ethernet interface:

```
switch# show interface ethernet 1/1
  Ethernet1/1 is up
  Hardware is 1000/10000 Ethernet, address is 000d.eca3.5f08 (bia 000d.eca3.5f08)
  MTU 1500 bytes, BW 10000000 Kbit, DLY 10 usec,
     reliability 255/255, txload 190/255, rxload 192/255
  Encapsulation ARPA
  Port mode is trunk
  full-duplex, 10 Gb/s, media type is 1/10g
  Input flow-control is off, output flow-control is off
  Auto-mdix is turned on
  Rate mode is dedicated
  Switchport monitor is off
  Last clearing of "show interface" counters never
  5 minute input rate 942201806 bytes/sec, 14721892 packets/sec
  5 minute output rate 935840313 bytes/sec, 14622492 packets/sec
  Rx
    129141483840 input packets 0 unicast packets 129141483847 multicast packets
    0 broadcast packets 0 jumbo packets 0 storm suppression packets
    8265054965824 bytes
    0 No buffer 0 runt 0 Overrun
    0 crc 0 Ignored 0 Bad etype drop
    0 Bad proto drop
  Tx
    119038487241 output packets 119038487245 multicast packets
   0 broadcast packets 0 jumbo packets
    7618463256471 bytes
    0 output CRC 0 ecc
    0 underrun 0 if down drop     0 output error 0 collision 0 deferred
    0 late collision 0 lost carrier 0 no carrier
    0 babble
```

```
    0 Rx pause 8031547972 Tx pause 0 reset
```

This example shows how to display the physical Ethernet capabilities:

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

  MDIX:                 no
  FEX Fabric:           yes
```

This example shows how to display the physical Ethernet transceiver:

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

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

```
switch# show interface brief

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

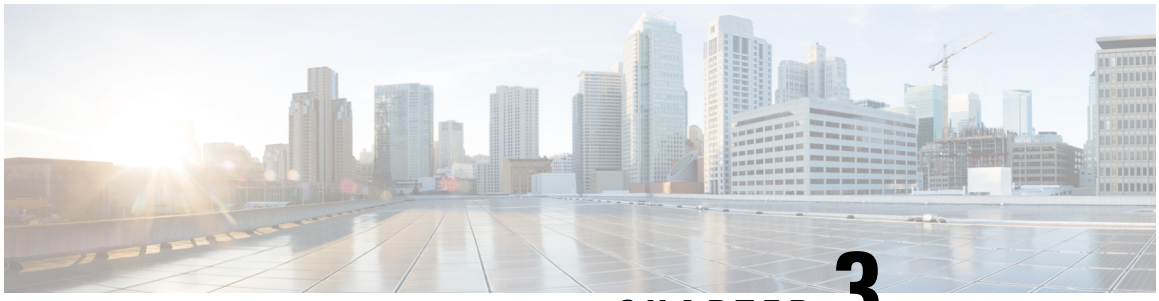
This example shows how to display the CDP neighbors:

```
switch# show cdp neighbors
Capability Codes: R - Router, T - Trans-Bridge, B - Source-Route-Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater,
```

```
                  V - VoIP-Phone, D - Remotely-Managed-Device,
                  s - Supports-STP-Dispute
Device ID              Local Intrfce   Hldtme  Capability  Platform      Port ID
d13-dist-1               mgmt0           148     S I      WS-C2960-24TC  Fas0/9
n5k(FLC12080012)         Eth1/5          8       S I s    N5K-C5020P-BA  Eth1/5
```

MIBs for Layer 2 Interfaces

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



CHAPTER 3

Configuring Layer 3 Interfaces

This chapter contains the following sections:

- Information About Layer 3 Interfaces, on page 39
- Guidelines and Limitations for Layer 3 Interfaces, on page 43
- Default Settings for Layer 3 Interfaces, on page 44
- SVI Autostate Disable, on page 44
- DHCP Client Discovery, on page 44
- MAC-Embedded IPv6 Address, on page 45
- Configuring Layer 3 Interfaces, on page 45
- Verifying the Layer 3 Interfaces Configuration, on page 59
- Triggering the Layer 3 Interface Consistency Checker, on page 60
- Monitoring Layer 3 Interfaces, on page 61
- Configuration Examples for Layer 3 Interfaces, on page 62
- Example of Changing VRF Membership for an Interface, on page 63
- Related Documents for Layer 3 Interfaces, on page 65
- MIBs for Layer 3 Interfaces, on page 65
- Standards for Layer 3 Interfaces, on page 65
- Feature History for Layer 3 Interfaces, on page 65

Information About Layer 3 Interfaces

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

Routed Interfaces

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

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

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

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

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

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

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

Subinterfaces

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

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

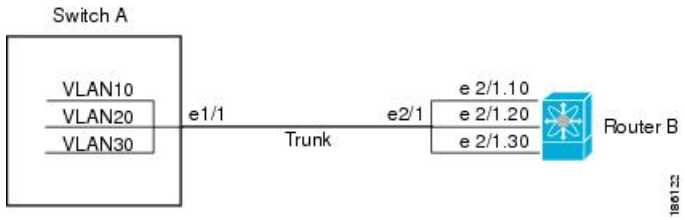
You create a subinterface with a name that consists of the parent interface name (for example, Ethernet 2/1) followed by a period and then by a number that is unique for that subinterface. For example, you could create a subinterface for Ethernet interface 2/1 named Ethernet 2/1.1 where .1 indicates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

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

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

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

Figure 2: Subinterfaces for VLANs



VLAN Interfaces

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

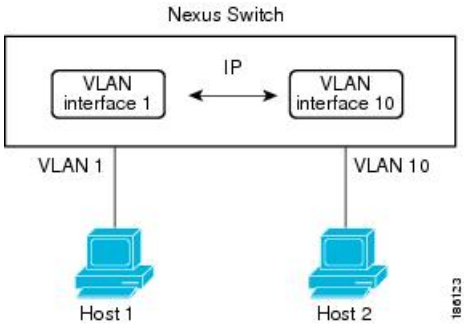
You must enable the VLAN network interface feature before you can configure it. The system automatically takes a checkpoint prior to disabling the feature, and you can roll back to this checkpoint. For information about rollbacks and checkpoints, see the System Management Configuration Guide for your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

Note You cannot delete the VLAN interface for VLAN 1.

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

The following figure shows two hosts connected to two VLANs on a device. You can configure VLAN interfaces for each VLAN that allows Host 1 to communicate with Host 2 using IP routing between the VLANs. VLAN 1 communicates at Layer 3 over VLAN interface 1and VLAN 10 communicates at Layer 3 over VLAN interface 10.

Figure 3: Connecting Two VLANs with VLAN Interfaces



Changing VRF Membership for an Interface

When you enter the **vrf member** command under an interface, you receive an alert regarding the deletion of interface configurations and to notify the clients/listeners (such as CLI-Server) to delete configurations with respect to the interface.

Entering the **system vrf-member-change retain-l3-config** command enables the retention of the Layer 3 configuration when the VRF member changes on the interface. It does this by sending notification to the clients/listeners to store (buffer) the existing configurations, delete the configurations from the old vrf context, and reapply the stored configurations under the new VRF context.

Note When the **system vrf-member-change retain-l3-config** command is enabled, the Layer 3 configuration is not deleted and remains stored (buffered). When this command is not enabled (default mode), the Layer 3 configuration is not retained when the VRF member changes.

You can disable the retention of the Layer 3 configuration with the **no system vrf-member-change retain-l3-config** command. In this mode, the Layer 3 configuration is not retained when the VRF member changes.

Notes About Changing VRF Membership for an Interface

- Momentary traffic loss may occur when changing the VRF name.
- Only the configurations under the interface level are processed when the **system vrf-member-change retain-l3-config** command is enabled. You must manually process any configurations at the router level to accommodate routing protocols after a VRF change.
- The **system vrf-member-change retain-l3-config** command supports interface level configurations with:
  - Layer 3 configurations maintained by the CLI Server, such as **ip address** and **ipv6 address** (secondary) and all OSPF/ISIS/EIGRP CLIs available under the interface configuration.
  - HSRP
  - DHCP Relay Agent CLIs, such as **ip dhcp relay address** [**use-vrf**] and **ipv6 dhcp relay address** [**use-vrf**].
- For DHCP:
  - As a best practice, the client and server interface VRF should be changed one at a time. Otherwise, the DHCP packets cannot be exchanged on the relay agent.
  - When the client and server are in different VRFs, use the **ip dhcp relay address** [**use-vrf**] command to exchange the DHCP packets in the relay agent over the different VRFs.

Loopback Interfaces

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

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

IP Unnumbered

The IP unnumbered feature enables the processing of IP packets on a point to point (p2p) interface without explicitly configuring a unique IP address on it. This approach borrows an IP address from another interface and conserves address space on point to point links.

Any interface which conforms to the point to point mode can be used as an IP unnumbered interface. For 7.0(3)I3(1) and later, the IP unnumbered feature is supported only on Ethernet interfaces and sub-interfaces. The borrowed interface can only be a loopback interface and is known as the numbered interface.

A loopback interface is ideal as a numbered interface in that it is always functionally up. However, because loopback interfaces are local to a switch/router, the reachability of unnumbered interfaces first needs to be established through static routes or by using an interior gateway protocol, such as OSPF or ISIS.

Staring from 7.0(3)I5(1), IP unnumbered feature is supported on port channel interfaces and sub-interfaces. The borrowed interface can only be a loopback interface and is known as the numbered interface.

Tunnel Interfaces

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

Guidelines and Limitations for Layer 3 Interfaces

Layer 3 interfaces have the following configuration guidelines and limitations:

- When an IP unnumbered interface is configured, a loopback interface should be in the same VRF as the IP unnumbered interface.
- An admin-shutdown command on a loopback interface that is a numbered interface does not bring down the IP unnumbered interface. This means that the routing protocols running over the IP unnumbered interface continue to be up.
- The static routes running over the IP unnumbered interface should use pinned static routes.

Note The IP unnumbered interface through which the route is resolved needs to be specified.

- Medium p2p should be enabled for configuring the IP unnumbered feature.

- If you change a Layer 3 interface to a Layer 2 interface, Cisco NX-OS shuts down the interface, reenables the interface, and removes all configuration specific to Layer 3.
- If you change a Layer 2 interface to a Layer 3 interface, Cisco NX-OS shuts down the interface, reenables the interface, and deletes all configuration specific to Layer 2.
- 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.
- Cisco Nexus 3000 Series switches punt multicast Layer 2 traffic to the CPU if the Layer 3 MTU is not the same for all Layer 3 interfaces, and if the MTU QoS was changed to jumbo. All Layer 3 interfaces must have the same Layer 3 MTU to avoid this issue.

Default Settings for Layer 3 Interfaces

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

SVI Autostate Disable

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

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

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

DHCP Client Discovery

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

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

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

Limitations for Using DHCP Client Discovery on Interfaces

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

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

MAC-Embedded IPv6 Address

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

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

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

Configuring Layer 3 Interfaces

Configuring a Routed Interface

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface ethernet** *slot/port*	Enters interface configuration mode.

	Command or Action	Purpose
Step 3	switch(conifg-if)# **no switchport**	Configures the interface as a Layer 3 interface and deletes any configuration specific to Layer 2 on this interface. **Note** To convert a Layer 3 interface back into a Layer 2 interface, use the **switchport** command.
Step 4	switch(config-if)# [**ip**|**ipv6**]*ip-address/length*	Configures an IP address for this interface.
Step 5	(Optional) switch(config-if)# **medium** {**broadcast** | **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.
Step 6	(Optional) switch(config-if)# **show interfaces**	Displays the Layer 3 interface statistics.
Step 7	(Optional) switch(config-if)# **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 IPv4-routed Layer 3 interface:

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

Configuring a Subinterface

Before you begin

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

Procedure

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

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

Configuring the Bandwidth on an Interface

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface ethernet** *slot/port*	Enters interface configuration mode. The range for the *slot* is from 1 to 255. The range for the *port* is from 1 to 128.

	Command or Action	Purpose
Step 3	switch(conifg-if)# **bandwidth** [*value* | **inherit** [*value*]]	Configures the bandwidth parameter for a routed interface, port channel, or subinterface, as follows: • *value*—Size of the bandwidth in kilobytes. The range is from 1 to 10000000. • **inherit**—Indicates that all subinterfaces of this interface inherit either the bandwidth value (if a value is specified) or the bandwidth of the parent interface (if a value is not specified).
Step 4	(Optional) switch(config-if)# **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 Ethernet interface 2/1 with a bandwidth value of 80000:

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

Configuring a VLAN Interface

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **feature interface-vlan**	Enables VLAN interface mode.
Step 3	switch(config)# **interface vlan** *number*	Creates a VLAN interface. The *number* range is from 1 to 4094.
Step 4	switch(config-if)# [**ip** | **ipv6**] **address** *ip-address/length*	Configures an IP address for this interface.
Step 5	switch(config-if)# **no shutdown**	Brings the interface up administratively.
Step 6	(Optional) switch(config-if)# **show interface vlan** *number*	Displays the VLAN interface statistics. The *number* range is from 1 to 4094.

	Command or Action	Purpose
Step 7	(Optional) switch(config-if)# **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 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
```

Enabling Layer 3 Retention During VRF Membership Change

The following steps enable the retention of the Layer 3 configuration when changing the VRF membership on the interface.

Procedure

	Command or Action	Purpose

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

Step 2	**system vrf-member-change retain-l3-config**
	Example:
	`switch(config)# system vrf-member-change retain-l3-config`
	`Warning: Will retain L3 configuration when vrf member change on interface.`
		Enables Layer 3 configuration retention during VRF membership change.
		Note To disable the retention of the Layer 3 configuration, use the **no system vrf-member-change retain-l3-config** command.

Configuring a Loopback Interface

Before you begin

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface loopback** *instance*	Creates a loopback interface. The *instance* range is from 0 to 1023.
Step 3	switch(config-if)# [**ip** | **ipv6**] **address** *ip-address/length*	Configures an IP address for this interface.
Step 4	(Optional) switch(config-if)# **show interface loopback** *instance*	Displays the loopback interface statistics. The *instance* range is from 0 to 1023.
Step 5	(Optional) switch(config-if)# **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 create a loopback interface:

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

Configuring IP Unnumbered on an Ethernet Interface

You can configure the IP unnumbered feature on an ethernet interface.

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**
	Example:
	`switch(config)# interface ethernet 1/1`
	`switch(config-if)#`
	`switch(config)# interface port-channel 1/1`
	`switch(config-if)#`
		Enters interface configuration mode. Supports Ethernet and Port-channel

Step 3	**medium p2p** **Example:**	Configures the interface medium as point to point.

	Command or Action	Purpose
	`switch(config-if)# medium p2p`	
Step 4	**ip unnumbered** *type number* **Example:** `switch(config-if)# ip unnumbered loopback 100`	Enables IP processing on an interface without assigning an explicit IP address to the interface. *type* and *number* specify another interface on which the router has an assigned IP address. The interface specified cannot be another unnumbered interface. **Note** *type* is limited to **loopback**. (7.0(3)I3(1) and later)

Configuring OSPF for an IP Unnumbered Interface

You can configure OSPF for an IP unnumbered loopback interface.

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 1/20.1`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**encapsulation dot1Q** *vlan-id* **Example:** `switch(config-if)# encapsulation dot1Q 100`	Configures IEEE 802.1Q VLAN encapsulation on the subinterface. The range is from 2 to 4093.
Step 4	**medium p2p** **Example:** `switch(config-if)# medium p2p`	Configures the interface medium as point to point.
Step 5	**ip unnumbered** *type number* **Example:** `switch(config-if)# ip unnumbered loopback 101`	Enables IP processing on an interface without assigning an explicit IP address to the interface. *type* and *number* specify another interface on which the router has an assigned IP address. The interface specified cannot be another unnumbered interface.

	Command or Action	Purpose
		Note *type* is limited to **loopback**. (7.0(3)I3(1) and later)
Step 6	(Optional) **ip ospf authentication** **Example:** `switch(config-if)# ip ospf authentication`	Specifies the authentication type for interface.
Step 7	(Optional) **ip ospf authentication-key** *password* **Example:** `switch(config-if)# ip ospf authentication 3 b7bdf15f62bbd250`	Specifies the authentication password for OSPF authentication.
Step 8	**ip router ospf** *instance* **area** *area-number* **Example:** `switch(config-if)# ip router ospf 100 area 0.0.0.1`	Configures routing process for IP on an interface and specifies an area. **Note** The **ip router ospf** command is required for both the unnumbered and the numbered interface.
Step 9	**no shutdown** **Example:** `switch(config-if)# no shutdown`	Brings up the interface (administratively).
Step 10	**interface loopback** *instance* **Example:** `switch(config)# interface loopback 101`	Creates a loopback interface. The range is from 0 to 1023.
Step 11	**ip address** *ip-address/length* **Example:** `switch(config-if)# 192.168.101.1/32`	Configures an IP address for the interface.
Step 12	**ip router ospf** *instance* **area** *area-number* **Example:** `switch(config-if)# ip router ospf 100 area 0.0.0.1`	Configures routing process for IP on an interface and specifies an area. **Note** The **ip router ospf** command is required for both the unnumbered and the numbered interface.

Configuring ISIS for an IP Unnumbered Interface

You can configure ISIS for an IP unnumbered loopback interface.

Procedure

	Command or Action	Purpose

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

Step 2	**feature isis** **Example:** `Switch(config)# feature isis`	Enables ISIS.
Step 3	**router isis** *area-tag* **Example:** `Switch(config)# router isis 100`	Assigns a tag to an IS-IS process and enters router configuration mode.
Step 4	**net** *network-entity-title* **Example:** `Switch(config-router)# net 49.0001.0100.0100.1001.00`	Configures the network entity title (NET) on the device.
Step 5	**end** **Example:** `Switch(config-router)# end`	Exit router configuration mode.
Step 6	**interface ethernet** *slot/port* **Example:** `switch(config)# interface ethernet 1/20.1`	Enters interface configuration mode.
Step 7	**encapsulation dot1Q** *vlan-id* **Example:** `switch(config-subif)# encapsulation dot1Q 100`	Configures IEEE 802.1Q VLAN encapsulation on the subinterface. The range is from 2 to 4093.
Step 8	**medium p2p** **Example:** `switch(config-subif)# medium p2p`	Configures the interface medium as point to point.
Step 9	**ip unnumbered** *type number* **Example:** `switch(config-if)# ip unnumbered loopback 101`	Enables IP processing on an interface without assigning an explicit IP address to the interface. *type* and *number* specify another interface on which the router has an assigned IP address. The interface specified cannot be another unnumbered interface.

	Command or Action	Purpose
		Note *type* is limited to **loopback**. (7.0(3)I3(1) and later)
Step 10	**ip router isis** *area-tag* **Example:** `switch(config-subif)# ip router isis 100`	Enables ISIS on the unnumbered interface.
Step 11	**no shutdown** **Example:** `switch(config-subif)# no shutdown`	Brings up the interface (administratively).

Assigning an Interface to a VRF

Before you begin

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface** *interface-typenumber*	Enters interface configuration mode.
Step 3	switch(conifg-if)#**vrf member** *vrf-name*	Adds this interface to a VRF.
Step 4	switch(config-if)# [**ip** | **ipv6**]*ip-address/length*	Configures an IP address for this interface. You must do this step after you assign this interface to a VRF.
Step 5	(Optional) switch(config-if)# **show vrf** [*vrf-name*] **interface** *interface-type number*	Displays VRF information.
Step 6	(Optional) switch(config-if)# **show interfaces**	Displays the Layer 3 interface statistics.
Step 7	(Optional) switch(config-if)# **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 add a Layer 3 interface to the VRF:

```
switch# configure terminal
switch(config)# interface loopback 0
switch(config-if)# vrf member RemoteOfficeVRF
```

```
switch(config-if)# ip address 209.0.2.1/16
switch(config-if)# copy running-config startup-config
```

Configuring an Interface MAC Address

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface ethernet** *slot/port*	Enters interface configuration mode.
Step 3	switch(config-if)# [**no**] **mac-address** *static router MAC address*	Configures the interface MAC address. The **no** form removes the configuration. You can enter the MAC address in any one of the four supported formats: • E.E.E • EE-EE-EE-EE-EE-EE • EE:EE:EE:EE:EE:EE • EEEE.EEEE.EEEE Do not enter any of the following invalid MAC addresses: • Null MAC address—0000.0000.0000 • Broadcast MAC address—FFFF.FFFF.FFFF • Multicast MAC address—0100.DAAA.ADDD
Step 4	switch(config-if)# **show interface ethernet** *slot/port*	(Optional) Displays all information for the interface.

Example

This example shows how to configure an interface MAC address:

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

Configuring a MAC-Embedded IPv6 Address

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface** *type slot/port*	Enters the interface configuration mode for the specified interface.
Step 3	switch(config-if)# **no switchport**	Configures the interface as a Layer 3 interface and deletes any configuration specific to Layer 2 on this interface. **Note** To convert a Layer 3 interface back into a Layer 2 interface, use the **switchport** command.
Step 4	switch(config-if)# **mac-address ipv6-extract**	Extracts the MAC address embedded in the IPv6 address configured on the interface. **Note** The MEv6 configuration is currently not supported with the EUI-64 format of IPv6 address.
Step 5	switch(config-if)# **ipv6 address** *ip-address/length*	Configures an IPv6 address for this interface.
Step 6	switch(config-if)# **ipv6 nd mac-extract** [**exclude nud-phase**]	Extracts the next-hop MAC address embedded in a next-hop IPv6 address. The **exclude nud-phase** option blocks packets during the ND phase only. When the **exclude nud-phase** option is not specified, packets are blocked during both ND and Neighbor Unreachability Detection (NUD) phases.
Step 7	(Optional) switch(config)# **show ipv6 icmp interface** *type slot/port*	Displays IPv6 Internet Control Message Protocol version 6 (ICMPv6) interface information.

Example

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

```
switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)# interface ethernet 1/3
switch(config-if)# no switchport
switch(config-if)# mac-address ipv6-extract
switch(config-if)# ipv6 address 2002:1::10/64
switch(config-if)# ipv6 nd mac-extract
switch(config-if)# show ipv6 icmp interface ethernet 1/3
```

```
ICMPv6 Interfaces for VRF "default"
Ethernet1/3, Interface status: protocol-up/link-up/admin-up
  IPv6 address: 2002:1::10
  IPv6 subnet:  2002:1::/64
  IPv6 interface DAD state:  VALID
  ND mac-extract : Enabled
  ICMPv6 active timers:
      Last Neighbor-Solicitation sent: 00:01:39
      Last Neighbor-Advertisement sent: 00:01:40
      Last Router-Advertisement sent: 00:01:41
      Next Router-Advertisement sent in: 00:03:34
  Router-Advertisement parameters:
      Periodic interval: 200 to 600 seconds
      Send "Managed Address Configuration" flag: false
      Send "Other Stateful Configuration" flag: false
      Send "Current Hop Limit" field: 64
      Send "MTU" option value: 1500
      Send "Router Lifetime" field: 1800 secs
      Send "Reachable Time" field: 0 ms
      Send "Retrans Timer" field: 0 ms
      Suppress RA: Disabled
      Suppress MTU in RA: Disabled
  Neighbor-Solicitation parameters:
      NS retransmit interval: 1000 ms
  ICMPv6 error message parameters:
      Send redirects: true
      Send unreachables: false
  ICMPv6-nd Statisitcs (sent/received):
      RAs: 3/0, RSs: 0/0, NAs: 2/0, NSs: 7/0, RDs: 0/0
      Interface statistics last reset: never
switch(config)#
```

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

```
switch# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
switch(config)# interface ethernet 1/5
switch(config-if)# no switchport
switch(config-if)# mac-address ipv6-extract
switch(config-if)# ipv6 address 2002:2::10/64
switch(config-if)# ipv6 nd mac-extract exclude nud-phase
switch(config-if)# show ipv6 icmp interface ethernet 1/5
ICMPv6 Interfaces for VRF "default"
Ethernet1/5, Interface status: protocol-up/link-up/admin-up
  IPv6 address: 2002:2::10
  IPv6 subnet:  2002:2::/64
  IPv6 interface DAD state:  VALID
  ND mac-extract : Enabled (Excluding NUD Phase)
  ICMPv6 active timers:
      Last Neighbor-Solicitation sent: 00:06:45
      Last Neighbor-Advertisement sent: 00:06:46
      Last Router-Advertisement sent: 00:02:18
      Next Router-Advertisement sent in: 00:02:24
  Router-Advertisement parameters:
      Periodic interval: 200 to 600 seconds
      Send "Managed Address Configuration" flag: false
      Send "Other Stateful Configuration" flag: false
      Send "Current Hop Limit" field: 64
      Send "MTU" option value: 1500
      Send "Router Lifetime" field: 1800 secs
      Send "Reachable Time" field: 0 ms
```

```
      Send "Retrans Timer" field: 0 ms
      Suppress RA: Disabled
      Suppress MTU in RA: Disabled
  Neighbor-Solicitation parameters:
      NS retransmit interval: 1000 ms
  ICMPv6 error message parameters:
      Send redirects: true
      Send unreachables: false
  ICMPv6-nd Statisitcs (sent/received):
      RAs: 6/0, RSs: 0/0, NAs: 2/0, NSs: 7/0, RDs: 0/0
      Interface statistics last reset: never
switch(config-if)#
```

Configuring SVI Autostate Disable

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

Procedure

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

Example

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

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

Configuring a DHCP Client on an Interface

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface ethernet** *type slot/port* | **mgmt** *mgmt-interface-number* | **vlan** *vlan id*	Creates a physical Ethernet interface, a management interface, or a VLAN interface. The range of *vlan id* is from 1 to 4094.
Step 3	switch(config-if)# [**no**] **ip** | **ipv6 address dhcp**	Requests the DHCP server for an IPv4 or IPv6 address. The **no** form of this command removes any address that was acquired.
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 the IP address of a DHCP client on an SVI:

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

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

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

Verifying the Layer 3 Interfaces Configuration

Use one of the following commands to verify the configuration:

Command	Purpose
show interface ethernet *slot/port*	Displays the Layer 3 interface configuration, status, and counters (including the 5-minute exponentially decayed moving average of inbound and outbound packet and byte rates).

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

Triggering the Layer 3 Interface Consistency Checker

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

Procedure

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

Example

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

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

Monitoring Layer 3 Interfaces

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

Command	Purpose
load-interval *seconds* | **counter** {**1** | **2** | **3**} *seconds*	Sets three different sampling intervals to bit-rate and packet-rate statistics. The range is from 5 seconds to 300 seconds.
show interface ethernet *slot*/*port* **counters**	Displays the Layer 3 interface statistics (unicast, multicast, and broadcast).
show interface ethernet *slot*/*port* **counters brief** *load-interval-id*	Displays the Layer 3 interface input and output counters. The load interval ID specifies a single load interval ID to display the input and output rates. The load interval ID ranges between 1 and 3.
show interface ethernet *slot*/*port* **counters detailed** [**all**]	Displays the Layer 3 interface statistics. You can optionally include all 32-bit and 64-bit packet and byte counters (including errors).
show interface ethernet *slot*/*port* **counters error**	Displays the Layer 3 interface input and output errors.
show interface ethernet *slot*/*port* **counters snmp**	Displays the Layer 3 interface counters reported by SNMP MIBs. You cannot clear these counters.
show interface ethernet *slot*/*port*.*number* **counters**	Displays the subinterface statistics (unicast, multicast, and broadcast).

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

Configuration Examples for Layer 3 Interfaces

This example shows how to configure Ethernet subinterfaces:

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

This example shows how to configure a VLAN interface:

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

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

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

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

This example shows how to configure a loopback interface:

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

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

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

Example of Changing VRF Membership for an Interface

- Enable Layer 3 configuration retention when changing VRF membership.

```
switch# configure terminal
switch(config)# system vrf-member-change retain-l3-config

Warning: Will retain L3 configuration when vrf member change on interface.
```

- Verify Layer 3 retention.

```
switch# show running-config | include vrf-member-change

system vrf-member-change retain-l3-config
```

- Configure the SVI interface with Layer 3 configuration as VRF "blue".

```
switch# configure terminal
switch(config)# show running-config interface vlan 2002

interface Vlan2002
description TESTSVI
no shutdown
mtu 9192
vrf member blue
no ip redirects
ip address 192.168.211.2/27
ipv6 address 2620:10d:c041:12::2/64
ipv6 link-local fe80::1
ip router ospf 1 area 0.0.0.0
ipv6 router ospfv3 1 area 0.0.0.0
hsrp version 2
hsrp 2002
preempt delay minimum 300 reload 600
priority 110 forwarding-threshold lower 1 upper 110
ip 192.168.211.1
hsrp 2002 ipv6
preempt delay minimum 300 reload 600
priority 110 forwarding-threshold lower 1 upper 110
ip 2620:10d:c041:12::1
```

- Change the SVI interface VRF to "red".

```
switch# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.
switch(config)# interface vlan 2002
switch(config-if)# vrf member red

Warning: Retain-L3-config is on, deleted and re-added L3 config on interface Vlan2002
```

- Verify SVI interface after VRF change.

```
switch# configure terminal
switch(config)# show running-config interface vlan 2002

interface Vlan2002
description TESTSVI
no shutdown
mtu 9192
vrf member red
no ip redirects
ip address 192.168.211.2/27
ipv6 address 2620:10d:c041:12::2/64
ipv6 link-local fe80::1
ip router ospf 1 area 0.0.0.0
ipv6 router ospfv3 1 area 0.0.0.0
hsrp version 2
hsrp 2002
preempt delay minimum 300 reload 600
priority 110 forwarding-threshold lower 1 upper 110
ip 192.168.211.1
hsrp 2002 ipv6
preempt delay minimum 300 reload 600
priority 110 forwarding-threshold lower 1 upper 110
ip 2620:10d:c041:12::1
```

Note

- When changing the VRF, the Layer 3 configuration retention affects:
  - Physical Interface
  - Loopback Interface
  - SVI Interface
  - Sub-interface
  - Tunnel Interface
  - Port-Channel
- When changing the VRF, the existing Layer 3 configuration is deleted and reapplied. All routing protocols, such as OSPF/ISIS/EIGRP/HSRP, go down in the old VRF and come up in the new VRF.
- Direct/Local IPv4/IPv6 addresses are removed from the old VRF and installed in the new VRF.
- Some traffic loss might occur during the VRF change.

Related Documents for Layer 3 Interfaces

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

MIBs for Layer 3 Interfaces

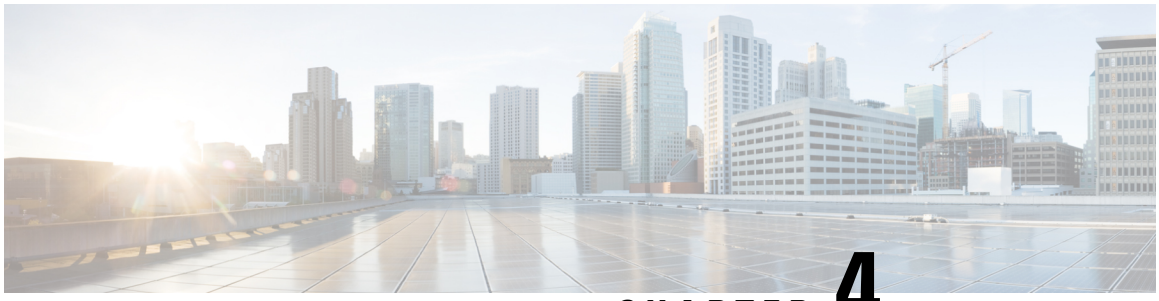
MIB	MIB Link
CISCO-IF-EXTENSION-MIB	To locate and download MIBs, go to the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml
ETHERLIKE-MIB	

Standards for Layer 3 Interfaces

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

Feature History for Layer 3 Interfaces

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



CHAPTER 4

Configuring Port Channels

This chapter contains the following sections:

- Information About Port Channels, on page 67
- Configuring Port Channels, on page 76
- Verifying Port Channel Configuration, on page 86
- Triggering the Port Channel Membership Consistency Checker, on page 86
- Verifying the Load-Balancing Outgoing Port ID , on page 87
- Feature History for Port Channels, on page 88
- Port Profiles, on page 88
- Configuring Port Profiles, on page 89
- Creating a Port Profile, on page 89
- Entering Port-Profile Configuration Mode and Modifying a Port Profile, on page 90
- Assigning a Port Profile to a Range of Interfaces, on page 91
- Enabling a Specific Port Profile, on page 92
- Inheriting a Port Profile, on page 93
- Removing a Port Profile from a Range of Interfaces, on page 93
- Removing an Inherited Port Profile, on page 94

Information About Port Channels

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

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

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

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

Related Topics

LACP Overview, on page 73

Understanding Port Channels

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

You can collect ports into a static port channel or you can enable the Link Aggregation Control Protocol (LACP). Configuring port channels with LACP requires slightly different steps than configuring static port channels. For information on port channel configuration limits, see the *Verified Scalability* document for your platform. For more information about load balancing, see Load Balancing Using Port Channels, on page 70.

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

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

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

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

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

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

Compatibility Requirements

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

The compatibility check includes the following operational attributes:

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

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

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

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

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

- Bandwidth
- MAC address
- Spanning Tree Protocol

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

- Description
- CDP
- LACP port priority
- Debounce

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

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

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

Load Balancing Using Port Channels

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

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

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

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

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

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

Table 3: Port Channel Load-Balancing Criteria

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

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

Regardless of the load-balancing algorithm configured, 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 This does not apply to Cisco Nexus 3500 Series switches.

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

Note Only the default load-balancing methods are currently supported based on src-dst ip and l4 ports for IP packets and src-dst mac for non-ip packets on the Cisco Nexus 34180YC and 3464C switches

Resilient Hashing

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

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

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

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

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

Hashing for NVGRE Traffic

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

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

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

Symmetric Hashing

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

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

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

Only the following load-balancing algorithms support symmetric hashing:

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

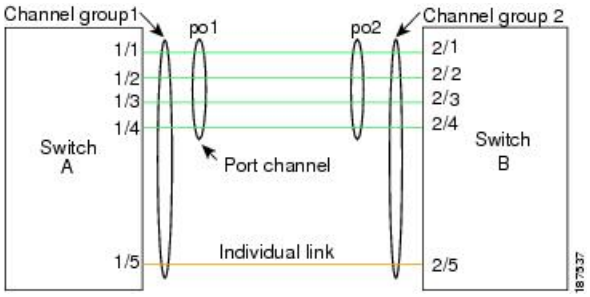
Understanding LACP

LACP Overview

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

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

Figure 4: Individual Links Combined into a Port Channel



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

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

You cannot disable LACP while any LACP configurations are present.

LACP ID Parameters

LACP uses the following parameters:

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

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

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

Channel Modes

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

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

The following table describes the channel modes.

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

Channel Mode	Description
passive	LACP mode that places a port into a passive negotiating state, in which the port responds to LACP packets that it receives but does not initiate LACP negotiation.

Channel Mode	Description
active	LACP mode that places a port into an active negotiating state, in which the port initiates negotiations with other ports by sending LACP packets.
on	All static port channels, that is, that are not running LACP, remain in this mode. If you attempt to change the channel mode to active or passive before enabling LACP, the device returns an error message. You enable LACP on each channel by configuring the interface in that channel for the channel mode as either active or passive. When an LACP attempts to negotiate with an interface in the on state, it does not receive any LACP packets and becomes an individual link with that interface; it does not join the LACP channel group. The **no lacp suspend-individual** configuration is supported by default on Cisco Nexus 3000 switches.

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

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

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

LACP Marker Responders

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

LACP-Enabled and Static Port Channel Differences

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

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

Configurations	Port Channels with LACP Enabled	Static Port Channels
Protocol applied	Enable globally.	Not applicable.

Configurations	Port Channels with LACP Enabled	Static Port Channels
Channel mode of links	Can be either: • Active • Passive	Can only be On.

LACP Port Channel Minimum Links and MaxBundle

A port channel aggregates similar ports to provide increased bandwidth in a single manageable interface. The introduction of the minimum links and MaxBundle feature further refines LACP port-channel operation and provides increased bandwidth in one manageable interface.

The LACP port channel MinLinks feature does the following:

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

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

Configuring Port Channels

Creating a Port Channel

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

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

Note Channel member ports cannot be a source or destination SPAN port.

Procedure

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

Example

This example shows how to create a port channel:

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

Adding a Port to a Port Channel

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

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface** *type slot*/*port*	Specifies the interface that you want to add to a channel group and enters the interface configuration mode.
Step 3	(Optional) switch(config-if)# **switchport mode trunk**	Configures the interface as a trunk port.
Step 4	(Optional) switch(config-if)# **switchport trunk** {**allowed vlan** *vlan-id* | **native vlan** *vlan-id*}	Configures necessary parameters for a trunk port.

	Command or Action	Purpose
Step 5	switch(config-if)# **channel-group** *channel-number*	Configures the port in a channel group and sets the mode. The channel-number range is from 1 to 4096. Cisco NX-OS creates the port channel associated with this channel group if the port channel does not already exist. This is called implicit port channel creation.
Step 6	(Optional) switch(config-if)# **no channel-group**	Removes the port from the channel group. The port reverts to its original configuration.

Example

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

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

Configuring Load Balancing Using Port Channels

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

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

Procedure

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

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

Example

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

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

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

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

Enabling LACP

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

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

Procedure

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

Example

This example shows how to enable LACP:

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

Configuring the Channel Mode for a Port

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

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

Before you begin

Ensure that you have enabled the LACP feature.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface** *type slot*/*port*	Specifies the interface to configure, and enters the interface configuration mode.
Step 3	switch(config-if)# **channel-group** *channel-number* [**force**] [**mode** {**on** | **active** | **passive**}]	Specifies the port mode for the link in a port channel. After LACP is enabled, you configure each link or the entire channel as active or passive.

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

Example

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

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

Configuring LACP Port Channel MinLinks

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

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface port-channel** *number*	Specifies the interface to configure.
Step 3	switch(config-if)# [**no**] **lacp min-links** *number*	Configures the number of minimum links. The default value for *number* is 1. The range is from 1 to 16. **Note** Starting with Release 7.0(3)I2(1), the maximum number of supported LACP min-links is 16. Use the **no** form of this command to disable this feature.
Step 4	(Optional) switch(config)# **show running-config interface port-channel** *number*	Displays the port channel configuration of the interface.

Example

This example shows how to configure the minimum number of links that must be up for the bundle as a whole to be labeled *up*:

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

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 an interface to configure.

Step 3	**lacp max-bundle** *number* **Example:** `switch(config-if)# lacp max-bundle <number>`	Configures the maximum number of active bundled LACP ports that are allowed in a port channel. 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 configuration for the interface.

Example

This example shows how to configure the maximum number of active bundled LACP ports:

```
switch# configure terminal
switch# interface port-channel 3
switch (config-if)# lacp max-bundle 3
switch (config-if)# show running-config interface port-channel 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.

Before you begin

Ensure that you have enabled the LACP feature.

Procedure

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

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 and System ID

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

Before you begin

Ensure that you have enabled the LACP feature.

Procedure

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

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

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

Before you begin

Ensure that you have enabled the LACP feature.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface** *type slot*/*port*	Specifies the interface to configure, and enters the interface configuration mode.
Step 3	switch(config-if)# **lacp port-priority** *priority*	Configures the port priority for use with LACP. Valid values are 1 through 65535, and higher numbers have lower priority. The default value is 32768.

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
```

Verifying Port Channel Configuration

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

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

Triggering the Port Channel Membership Consistency Checker

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

Procedure

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

Example

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

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

Verifying the Load-Balancing Outgoing Port ID

Command Guidelines

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

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

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

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

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

Example

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

```
switch# show port-channel load-balance forwarding-path interface port-channel 10 vlan 1
dst-ip 1.225.225.225 src-ip 1.1.10.10 src-mac aa:bb:cc:dd:ee:ff
l4-src-port 0 l4-dst-port 1
```

```
Missing params will be substituted by 0's. Load-balance Algorithm on switch: source-dest-port
 crc8_hash:204 Outgoing port id: Ethernet 1/1 Param(s) used to calculate load balance:
dst-port: 0
src-port: 0
dst-ip: 1.225.225.225
src-ip: 1.1.10.10
dst-mac: 0000.0000.0000
src-mac: aabb.ccdd.eeff
```

Feature History for Port Channels

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

Port Profiles

- 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.

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.

Configuring Port Profiles

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:
  - .
  - -
  - _

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.

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*	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.

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.
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.

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.
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.

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.

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.

	Command or Action	Purpose
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.

Procedure

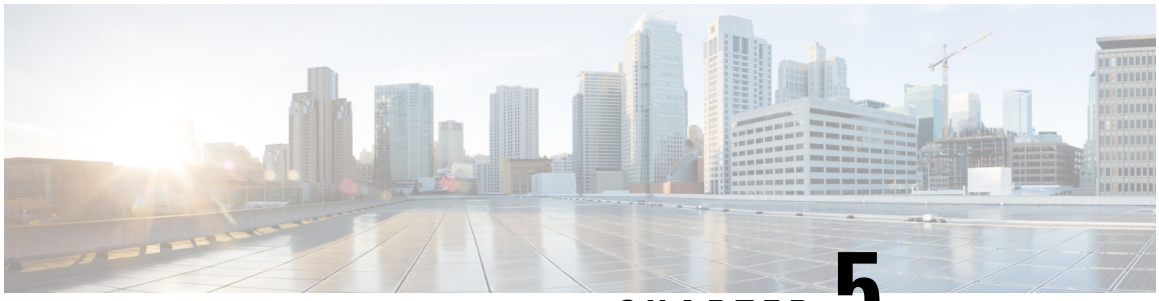
	Command or Action	Purpose
Step 1	**configure terminal**	Enters the global configuration mode.
Step 2	**port-profile** *name*	Enters the port-profile configuration mode for the specified port profile.
Step 3	**no inherit port-profile** *name*	Removes an inherited port profile from this port profile.
Step 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)#
```



CHAPTER 5

Configuring IP Tunnels

This chapter contains the following sections:

- Information About IP Tunnels, on page 97
- Prerequisites for IP Tunnels, on page 98
- Guidelines and Limitations for IP Tunnels, on page 98
- Default Settings for IP Tunneling, on page 102
- Configuring IP Tunnels, on page 102
- Verifying the IP Tunnel Configuration, on page 110
- Configuration Examples for IP Tunneling, on page 111
- Related Documents for IP Tunnels, on page 111
- Standards for IP Tunnels, on page 111
- Feature History for Configuring IP Tunnels, on page 112

Information About IP Tunnels

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

IP tunnels consists of the following three main components:

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

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

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

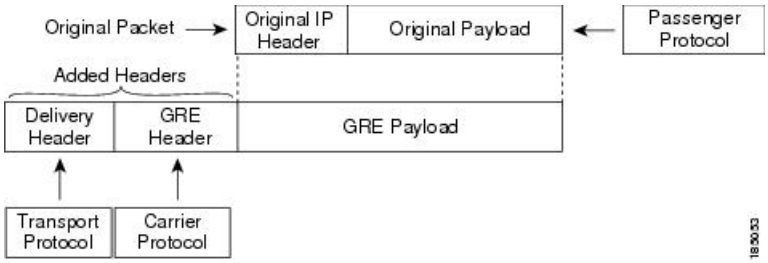
You must enable the tunnel feature before you can configure it.

GRE Tunnels

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

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

Figure 5: GRE PDU



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

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

Multi-Point IP-in-IP Tunnel Decapsulation

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

Prerequisites for IP Tunnels

IP tunnels have the following prerequisites:

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

Guidelines and Limitations for IP Tunnels

IP tunnels have the following configuration guidelines and limitations:

- Guidelines for **source-direct** and **ipv6ipv6-decapsulate-any** options for tunnels:

-
-
- The **tunnel source direct** command is supported only when an administrator uses the IP-in-IP decapsulation to source route the packets through the network. The source-direct tunnel is always operationally *Up* unless it is administratively shut down. The directly connected interfaces are identified using the **show ip route direct** command.
- The **tunnel source direct** command is supported only on decapsulate-any tunnel modes, for example, **tunnel mode ipip decapsulate-any** and **tunnel mode ipv6ipv6 decapsulate-any**.
- Auto-recovery is not supported for source-direct.
- For **ipv6ipv6 decapsulate-any**, inter-VRF is not supported. The tunnel interface VRF (iVRF) and tunnel transport or forwarding VRF (fVRF) must be the same. Only one decapsulate-any tunnel (irrespective of VRF) can be present in Cisco Nexus 3000 Series switches.
- To enable IPv6 on ipv6ipv6 decap-any tunnel interface, you must configure a valid IPv6 address or configure the ipv6 address using use-link-local-only CLI command under the interface tunnel interface.

- The hardware limitations on a source direct tunnel are as follows:
  - Source direct tunnel supports Cisco Nexus 3000 Series switches with Network Forwarding Engine (NFE), Application Spine Engine (ASE), and Leaf Spine Engine (LSE). There are limitations in cases of scaled SIP (number of total IP/IPv6 addresses on the interfaces (L3, sub-interface, PC, PC-sub interfaces, loopback, SVI, and any secondary IP/IPv6 addresses.)

    See the following sample use cases.
    - Use Case 1: Non-deterministic behavior of which SIP gets installed if the number of IP/IPv6 interface scale is more.

      Both the switches have 512 entries for tunnel SIP. With tunnel source, direct any IP or IPv6 address w.r.t **ipip or ipv6ipv6 decap any** with tunnel source gets installed in the above table.

      The insertion of these entries is on a first come first serve basis without any CLI command to control which interface IP addresses get installed. If the system has more number of IP/IPv6 interfaces to be installed, the behavior is non-deterministic (The behavior can change across reload with interface flaps.)
    - Use Case 2: The scale numbers are different in both switches and each has its own advantages and disadvantages.

      IPv4 individual scale can be more (up to 512) in case of switches with NFE. In the switches with ASE and LSE, the IPv4 individual scale can be 256 but it is shared with IPv6. If the user plans to configure both v4 and v6 decap any tunnel in the same system, the scale numbers for the switches with NFE for individual IPv4 and IPv6 cannot be guaranteed. However, the scale numbers for the switches with ASE and LSE for individual IPv4 and IPv6 are guaranteed. These is no CLI command to change these pre-carved scale numbers, for example, allocating X for IPv4 and Y for IPv6.

      Whenever the tunnel decap table gets full, the TABLE_FULL error is displayed. If an entry gets deleted after the table is full, the table full error is cleared.

      If the tunnel-decap-table is full, the user gets a syslog similar to as follows:

      ```
      2017 Apr 26 10:10:51 switch %$ VDC-1 %$
      ```

```
%IPFIB-2-FIB_HW_IP_TUNNEL_DECAP_TABLE_FULL:
IP TUNNEL decap hardware table full. IP tunnel decapsulation may not work for
some GRE/IPinIP traffic
```

If the table is full and if an entry is deleted from the table because of an interface being operationally down or removal of IP address, the clear syslog for the table is displayed. Deleting of a tunnel removes all the entries that are added as part of that tunnel.

```
 2002 Sep 26 10:11:37 switch %$ VDC-1 %$
%IPFIB-2-FIB_HW_IP_TUNNEL_DECAP_TABLE_FULL_CLRD: IP TUNNEL decap hardware table

full exception cleared
```

- ***Table 6: Scale Numbers***

Commands	Switches with NFE: Table size 512, v4 takes 1 entry, v6 takes 4 entries	Switches with ASE and LSE: Table size 512, v4 takes 1 entry, v6 takes 2 entries (paired index)
IPIP decap any with tunnel source direct	Shared between v4 and v6, v6 takes 4 entries v4 + 4 *v6 =512 Maximum entries can be 512 with no v6 entries	Dedicated 256
IPv6IPv6 decap any with tunnel source direct	Shared between v4 and v6, v6 takes 4 entries v4 + 4 *v6 =512 Maximum entries can be 128 with no v4 entries	Dedicated 128

- Use Case 3: Auto-recovery is not supported.

  If any entry does not get installed in the hardware due to exhaustion of above table, removal of an already installed IP/IPv6 from interfaces does not automatically trigger the addition of the failed SIP in the table though the table has space now. You need to flap the tunnel interface or IP interface to get them installed.

  However, if an entry does not get installed in the hardware due to a duplicate entry (if there was already a **decap-any** with one source present and now the **source direct tunnel** CLI command is configured, there is a duplicate entry for the prior source configured) that was taken care of by removing the entry only when both the tunnels get deleted.

-
```
interface loopback0
  ip address 2001:0:0:4::1/128
!
interface Tunnel 1
  ipv6 address use-link-local-only
  tunnel mode tunnel mode ipv6ipv6 decapsulate-any
  tunnel source loopback0
  description IPinIP Decapsulation Interface
```

```
mtu 1476
no shutdown
```

- Cisco NX-OS software supports the GRE header defined in IETF RFC 2784. Cisco NX-OS software does not support tunnel keys and other options from IETF RFC 1701.
- The Cisco Nexus device supports the following maximum number tunnels:
  - GRE and IP-in-IP regular tunnels-8 tunnels
  - Multipoint IP-in-IP tunnels-32 tunnels
- Each tunnel will consume one Equal Cost Multipath (ECMP) adjacency.
- The Cisco Nexus device does not support the following features:
  - Path maximum transmission unit (MTU) discovery
  - Tunnel interface statistics
  - Access control lists (ACLs)
  - Unicast reverse path forwarding (URPF)
  - Multicast traffic and associated multicast protocols such as Internet Group Management Protocol (IGMP) and Protocol Independent Multicast (PIM)
- Cisco NX-OS software does not support the Web Cache Control Protocol (WCCP) on tunnel interfaces.
- Cisco NX-OS software supports only Layer-3 traffic.
- Cisco NX-OS software supports ECMP across tunnels and ECMP for tunnel destination.
- IPv6-in-IPv6 tunnels is not supported.
- Limited control protocols, such as Border Gateway Protocol (BGP), Open Shortest Path First (OSPF), and Enhanced Interior Gateway Routing Protocol (EIGRP), are supported for GRE tunnels.
- Starting with Release 6.0(2)U5(1), Cisco Nexus 3000 Series switches drop all the packets when the tunnel is not configured. The packets are also dropped when the tunnel is configured but the tunnel interface is not configured or the tunnel interface is in shut down state.

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

  Decapsulate Tunnel (Source only) - Cisco Nexus 3000 Series switches decapsulate all IP-in-IP packets destined to it when the command **feature tunnel** is configured and there is an operational tunnel interface configured with the tunnel source address that matches the incoming packets' outer destination addresses. If there is not a source packet match or if the interface is in shutdown state, the packet is dropped.
- Starting with Release 6.0(2)U6(1), Cisco Nexus 3000 Series switches support IPv6 in IPv4 with GRE header only. The new control protocols that are supported on the tunnel are:
  - BGP with v6
  - OSPFv3

  - EIGRP over v6

- GRE v4/v6 tunnel configuration is supported only in the default routing mode. It does not support the multicast traffic or multicast protocols, for example, IGMP/PIM. It does not support ACL/QoS policies. It supports a maximum of 8 tunnels in the switch, whether they are all IPinIP or GRE; or any combination of both. The packets that are sent/received over the tunnel and that are destined for the switch, are not counted in the tunnel statistics.
- The Cisco Nexus 3000 Series switches ASIC supports the GRE encapsulation and decapsulation in the hardware.
- On the encapsulation side, the Cisco Nexus 3000 Series switches performs a single lookup in the hardware.
- Since Cisco Nexus 3000 Series switches perform a single lookup in the hardware, the software has to keep the hardware information up-to-date with any changes related to the second lookup, for example, the tunnel destination adjacency.
- On the decapsulation side, the Cisco Nexus 3000 Series switches have a separate table to perform the outer IP header lookup and it does not need an ACL for the same.
- RFC5549 is not supported over tunnels.

Default Settings for IP Tunneling

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

Table 7: Default IP Tunnel Parameters

Parameters	Default
Tunnel feature	Disabled

Configuring IP Tunnels

Enabling Tunneling

Before you begin

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **feature tunnel**	Enables the tunnel feature on the switch.

	Command or Action	Purpose
Step 3	switch(config)# **exit**	Returns to configuration mode.
Step 4	switch(config)# **show feature**	Displays the tunnel feature on the switch.
Step 5	(Optional) switch# **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 enable the tunnel feature:

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

Creating a Tunnel Interface

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

Before you begin

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

Ensure that you have enabled the tunneling feature.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# [**no**] **interface tunnel** *number*	Creates a new tunnel interface.
Step 3	switch(config)# **tunnel mode** {**gre ip** | **ipip** {**ip** | **decapsulate-any**}}	Sets this tunnel mode to GRE, ipip, or ipip decapsulate-only. 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.

	Command or Action	Purpose
Step 4	switch(config)# **tunnel source** {*ip address* | *interface-name*}	Configures the source address for this IP tunnel.
Step 5	switch(config)# **tunnel destination** {*ip address* | *host-name*}	Configures the destination address for this IP tunnel.
Step 6	(Optional) switch(config)# **tunnel use-vrf** *vrf-name*	Uses the configured VRF instance to look up the tunnel IP destination address.
Step 7	(Optional) switch(config)# **show interface tunnel** *number*	Displays the tunnel interface statistics.
Step 8	(Optional) switch# **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 create a tunnel interface:

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

Configuring a Tunnel Interface

The **tunnel mode ipv6ipv6 decapsulate-any** command supports IPv6 payload over IPv6 transport (IPv6inIPv6 packets). You can configure IP-in-IP tunnel decapsulation on directly connected IP addresses (for example, physical interface, port-channel, loopback, and SVI) using the new **tunnel source direct** CLI command.

Before you begin

Ensure that you have enabled the tunneling feature.

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.

	Command or Action	Purpose
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	(Optional) **tunnel mode ipv6ipv6 decapsulate-any**	Supports IPv6 payload over IPv6 transport (IPv6inIPv6 packets) This step is applicable for IPv6 networks only.
Step 5	**tunnel source direct**	Configures IP-in-IP tunnel decapsulation on any directly connected IP addresses. this option is now supported only when the IP-in-IP decapsulation is used to source route the packets through the network.
Step 6	**show interfaces tunnel** *number* **Example:** `switch(config-if)# show interfaces tunnel 1`	(Optional) Displays the tunnel interface statistics.
Step 7	**mtu** *value*	Sets the maximum transmission unit (MTU) of IP packets sent on an interface. The range is from 64 to 9192 units.
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 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
```

This example shows how to configure IP-in-IP tunnel decapsulation on directly connected IP addresses:

```
switch# configure terminal
switch(config)# interface tunnel 0
switch(config-if)# tunnel mode ipip ip
switch(config-if)# tunnel source direct
switch(config-if)# description IPinIP Decapsulation Interface
switch(config-if)# no shut
```

This example shows how to configure IP-in-IP tunnel decapsulation on IPv6 enabled networks:

```
interface loopback0
  ip address 2001:0:0:4::1/128
!
interface Tunnel1
  tunnel mode ipip decapsulate-any ipv6
  tunnel source loopback0
  description IPinIP Decapsulation Interface
  mtu 1476
  no shutdown

show running-config interface tunnel 1
interface Tunnel1
  tunnel mode ipv6ipv6 decapsulate-any
  tunnel source direct
  no shutdown

show interface tunnel 1
Tunnel1 is up    Admin State: up
MTU 1460 bytes, BW 9 Kbit
Tunnel protocol/transport IPv6/DECAPANY/IPv6
Tunnel source - direct
Transport protocol is in VRF "default"
Tunnel interface is in VRF "default"
Last clearing of "show interface" counters never
Tx    0 packets output, 0 bytes    Rx    0 packets input, 0 bytes
```

Configuring a Tunnel Interface Based on Policy Based Routing

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

Before you begin

Ensure that you have enabled the tunneling feature.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# [**no**] **interface tunnel** *number*	Creates a new tunnel interface.
Step 3	switch(config)# **ip address** *ip address*	Configures an IP address for this interface.

	Command or Action	Purpose
Step 4	switch(config)# **route-map** *map-name*	Assigns a route map for IPv4 policy-based routing to the interface
Step 5	switch(config-route-map)# **match ip address access-list-name** *name*	Matches an IPv4 address against one or more IP access control lists (ACLs). This command is used for policy-based routing and is ignored by route filtering or redistribution.
Step 6	switch(config-route-map)# **set ip next-hop** *address*	Sets the IPv4 next-hop address for policy-based routing. To select tunnel interfaces, you must specify the Tunnel IP addresses as next-hop addresses. This command uses the first valid next-hop address if multiple addresses are configured. Use the **load-share** option to select ECMP across next-hop entries.

Example

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

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

Configuring a GRE Tunnel

GRE v6 tunnel is used to carry different types of packets over IPv6 transport. GREv6 tunnel carries only IPv4 payload. The tunnel CLIs are enhanced to select IPv6 tunnel and configure v6 tunnel source and destination.

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

Before you begin

Ensure that you have enabled the tunneling feature.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface tunnel** *number*	Enters a tunnel interface configuration mode.
Step 3	switch(config-if)# **tunnel mode** {**gre ip** | **ipip** {**ip** | **decapsulate-any**}}	Sets this tunnel mode to GRE, ipip, or ipip decapsulate-only.

	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	Required: switch(config-if)# **tunnel use-vrf** *vrf-name*	Configures tunnel VRF name.
Step 5	Required: switch(config-if)# **ipv6 address** *IPv6 address*	Configures the IPv6 address. **Note** The tunnel source and the destination addresses are still the same (IPv4 address.)
Step 6	(Optional) switch(config-if)# **show interface tunnel** *number*	Displays the tunnel interface statistics.
Step 7	switch(config-if)# **mtu** *value*	Sets the maximum transmission unit (MTU) of IP packets sent on an interface.
Step 8	(Optional) switch(config-if)# **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 displays how to configure IPv6 Payload over GRE v4 tunnel. Configure the tunnel source, destination, IPv4 address, IPv6 address, and perform the **no shut** command. Once the GREv4 tunnel is created, you can configure v4 or v6 route via the tunnel:

```
switch# configure terminal
switch(config)# interface tunnel 10
switch(config)# tunnel source 11.1.1.1
switch(config)# tunnel destination 11.1.1.2
switch(config-if)# tunnel mode gre ip
switch(config-if)# tunnel use-vrf red
switch(config-if)# ip address 10.1.1.1/24
switch(config-if)# ipv6 address 2:2::2/64
switch(config-if)# no shut

switch(config)# ip route 50.1.1.0/24 tunnel 10
switch(config)# ipv6 route 2000:100::/64 tunnel 10
```

This example shows how to view the GRE v4 tunnel interface 10 and display IPv4 and IPv6 routes:

```
switch(config)# show int tunnel 10
Tunnel 10 is up
    Admin State: up
    Internet address(es):
```

```
    10.1.1.1/24
    1010::1/64
    MTU 1476 bytes, BW 9 Kbit
    Tunnel protocol/transport GRE/IP
    Tunnel source 11.1.1.1, destination 11.1.1.2
    Transport protocol is in VRF "default"

switch#show ipv6 route
…
2000:100::/64, ubest/mbest: 1/0, attached
    *via Tunnel10, [1/0], 00:00:16, static

#show ip route
…
50.1.1.0/24, ubest/mbest: 1/0
    *via Tunnel10, [1/0], 00:03:33, static
```

This example displays how to configure IPv4 payload over GRE v6 tunnel. Configure the tunnel mode as GRE IPv6, tunnel v6 source and destination, IPv4 address, and perform the **no shut** command. Once the GREv6 tunnel is created, you can configure v4 route via the tunnel:

```
switch# configure terminal
switch(config)# interface tunnel 20
switch(config-if)# tunnel mode gre ipv6
switch(config)# tunnel source 1313::1
switch(config)# tunnel destination 1313::2
switch(config-if)# tunnel use-vrf red
switch(config-if)# ip address 20.1.1.1/24
switch(config-if)# no shut

switch(config)# ip route 100.1.1.0/24 tunnel 20
```

This example displays how to view the GREv6 tunnel interface 20:

```
show interface tunnel 20
Tunnel 20 is up
    Admin State: up
    Internet address is 20.1.1.1/24
    MTU 1456 bytes, BW 9 Kbit
    Tunnel protocol/transport GRE/IPv6
    Tunnel source 1313::1, destination 1313::2
    Transport protocol is in VRF "default"

#show ip route
…
100.1.1.0/24, ubest/mbest: 1/0
    *via Tunnel20, [1/0], 00:01:00, static

red10# show interface brief | grep Tunnel
Tunnel10                up         10.1.1.1/24       GRE/IP          1476
Tunnel20                up         20.1.1.1/24       GRE/IPv6        1456
```

This example shows how to create an ipip tunnel:

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

Assigning VRF Membership to a Tunnel Interface

You can add a tunnel interface to a VRF.

Before you begin

Ensure that you have enabled the tunneling feature.

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface tunnel** *number*	Enters interface configuration mode.
Step 3	switch(config)# **vrf member** *vrf-name*	Adds this interface to a VRF.
Step 4	switch(config)# **ip address** *ip-prefix/length*	Configures an IP address for this interface. You must do this step after you assign this interface to a VRF.
Step 5	(Optional) switch(config)# **show vrf** [*vrf-name*] **interface** *interface-type number*	Displays VRF information.
Step 6	(Optional) switch(config-if)# **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 add a tunnel interface to the VRF:

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

Verifying the IP Tunnel Configuration

Use the following commands to verify the configuration:

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

Command	Purpose
show interface tunnel *number* **description**	Displays the configured description of the tunnel interface.
show interface tunnel *number* **status**	Displays the operational status of the tunnel interface.
show interface tunnel *number* **status err-disabled**	Displays the error disabled status of the tunnel interface.

Configuration Examples for IP Tunneling

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

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

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

Related Documents for IP Tunnels

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

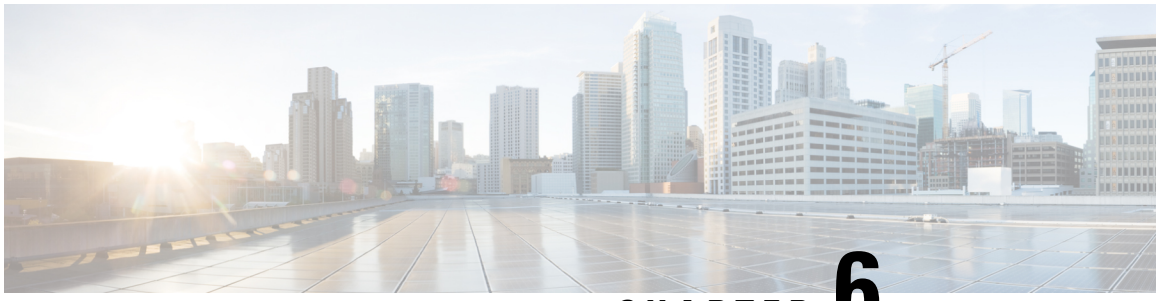
Standards for IP Tunnels

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

Feature History for Configuring IP Tunnels

Table 8: Feature History for Configuring IP Tunnels

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



CHAPTER 6

Configuring VXLANs

This chapter contains the following sections:

- Overview, on page 113
- Configuring VXLAN Traffic Forwarding, on page 122
- Verifying the VXLAN Configuration, on page 131
- Overview of IGMP Snooping Over VXLAN, on page 133
- Guidelines and Limitations for IGMP Snooping Over VXLAN, on page 133
- Configuring IGMP Snooping Over VXLAN, on page 133

Overview

VXLAN Overview

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

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

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

A VXLAN has the following benefits:

- Flexible placement of multitenant segments throughout the data center.

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

- Higher scalability to address more Layer 2 segments.

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

- Utilization of available network paths in the underlying infrastructure.

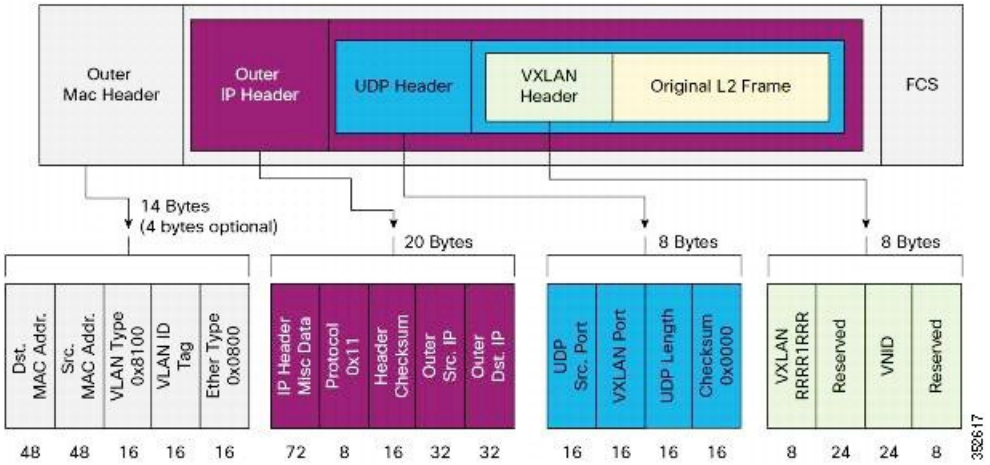
  VXLAN packets are transferred through the underlying network based on its Layer 3 header. It uses equal-cost multipath (ECMP) routing and link aggregation protocols to use all available paths.

VXLAN Encapsulation and Packet Format

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

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

Figure 6: VXLAN Packet Format



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

VXLAN Tunnel Endpoints

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

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

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

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

VXLAN Packet Forwarding Flow

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

VXLAN Implementation on Cisco Nexus 3100 Platform Switches

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

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

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

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

A VXLAN can forward these traffic types as follows:

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

Layer 2 Mechanisms for Unicast-Learned Traffic

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

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

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

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

VXLAN Layer 2 Gateway as a Transit Multicast Router

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

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

ECMP and LACP Load Sharing with VXLANs

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

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

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

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

Guidelines and Limitations for VXLANs

VXLAN has the following guidelines and limitations:

- The configuration of the multicast groups and Ingress Replication (IR) is not supported at the same time. You can configure and deploy either multicast groups or IR to deploy VXLAN.
- The **system vlan nve-overlay** CLI is not required in Cisco Nexus 3000 Series switches with certain types of BroadCom ASICs. Therefore, do not enable the **system vlan nve-overlay** CLI command.
- In VXLAN on vPC configuration, the packets from North VTEP are decapped on the primary vPC switch and they are sent to all ports in the VLAN/VN-segment and they are also forwarded on the multicast link

to the secondary vPC switch. Therefore, the NVE VNI counters are observed to increment for both Tx and Rx on the primary vPC switch, whereas the NVE VNI counters increment only for Rx on the secondary vPC switch.

- It is recommended that the summation of the number of the multicast groups and the OIFLs to be used in a scaled environment should not exclude 1024 which is the current range of the multicast VXLAN VP.
- Adjacencies are configured in different regions on an overlay or underlay network for different types of L3 interfaces based on whether or not the VxLAN, VNI or VFI are enabled on the interface. MAC rewrite does not happen if packets sent from a VFI enabled VLAN and hit an adjacency in an underlay network. So routing between VxLAN enabled VLANs and non-VxLAN enabled VLANs or L3 interfaces may fail.
- Starting with Release 7.0(3)I5(1), IGMP snooping is supported on VXLAN VLANs.
- VXLAN routing is not supported. The default Layer 3 gateway for VXLAN VLANs must be provisioned on a different device.

Note Starting with Cisco NX-OS Release 7.0(3)I4(1), VXLAN routing is supported for the Cisco Nexus 3100-V platform switches.

- Ensure that the network can accommodate an additional 50 bytes for the VXLAN header.
- Only one Network Virtualization Edge (NVE) interface is supported on a switch.
- Layer 3 VXLAN uplinks are not supported in a nondefault virtual and routing forwarding (VRF) instance.
- Only one VXLAN IP adjacency is possible per physical interface.
- Switched virtual interfaces (SVIs) are not supported on VXLAN VLANs.

Note Starting with Cisco NX-OS Release 7.0(3)I4(1), SVIs over VXLAN VLAN for routing are supported for the Cisco Nexus 3100-V platform switches.

- Switched Port Analyzer (SPAN) Tx for VXLAN-encapsulated traffic is not supported for the Layer 3 uplink interface.
- Access control lists (ACLs) and quality of service (QoS) for VXLAN traffic to access direction are not supported.
- SNMP is not supported on the NVE interface.
- Native VLANs for VXLAN are not supported.
- For ingress replication configurations, multiple VNIs can now have the same remote peer IP configured.
- The VXLAN source UDP port is determined based on the VNID and source and destination IP addresses.
- The UDP port configuration must be done before the NVE interface is enabled. If the UDP configuration must be changed while the NVE interface is enabled, you must shut down the NVE interface, make the UDP configuration change, and then reenable the NVE interface.

Note Starting with Cisco NX-OS Release 7.0(3)I4(1), the VXLAN UDP port is not configurable on the Cisco Nexus 3100-V platform switches.

- When a VN-Segment is mapped to a native VLAN, if traffic is sent on any normal VLAN on that port instead of getting switched in the VLAN, it gets forwarded in the VXLAN tunnel for the native VLAN.
- Starting with Cisco NX-OS Release 7.0(3)I6(1), VXLAN is supported on Cisco Nexus 3232C and 3264Q switches. Inter-VNI routing and IGMP snooping for VXLAN-enabled VLANs are not supported on Cisco Nexus 3232C and 3264Q switches.
- In VXLAN EVPN setup that has 2K VNI scale configuration, the control plane downtime takes more than 200 seconds. You must configure the graceful restart time as 300 seconds to avoid BGP flap.
- Starting with Cisco NX-OS Release 7.0(3)I7(1), FHRP over VXLAN is supported on Cisco Nexus 3100-V platform switches.
- Starting with Cisco NX-OS Release 7.0(3)I7(1), HSRP over VXLAN is supported on Cisco Nexus 3100-V platform switches.

FHRP Over VXLAN

Overview of FHRP over VXLAN

Overview of FHRP

Starting with Cisco NX-OS Release 7.0(3)I7(1), you can configure First Hop Redundancy Protocol (FHRP) over VXLAN on Cisco Nexus 3000 Series switches. The FHRP provides a redundant Layer 3 traffic path. It provides fast failure detection and transparent switching of the traffic flow. The FHRP avoids the use of the routing protocols on all the devices. It also avoids the traffic loss that is associated with the routing or the discovery protocol convergence. It provides an election mechanism to determine the next best gateway. Current FHRP supports HSRPv1, HSRPv2, VRRPv2, and VRRPv3.

FHRP over VXLAN

The FHRP serves at the Layer 3 VXLAN redundant gateway for the hosts in the VXLAN. The Layer 3 VXLAN gateway provides routing between the VXLAN segments and routing between the VXLAN to the VLAN segments. Layer 3 VXLAN gateway also serves as a gateway for the external connectivity of the hosts.

Guidelines and Limitations for FHRP Over VXLAN

See the following guidelines and limitations for configuring FHRP over VXLAN:

- When using FHRP with VXLAN, ARP-ETHER TCAM must be carved using the **arp-ether 256 double-wide** CLI command.
- Configuring FHRP over VXLAN is supported for both IR and multicast flooding of the FHRP packets. The FHRP protocol working does not change for configuring FHRP over VXLAN.
- The FHRP over VXLAN feature is supported for flood and learn only.
- For Layer 3 VTEPs in BGP EVPN, only anycast GW is supported.

- Starting with Cisco NX-OS Release 7.0(3)I7(1), FHRP over VXLAN is supported on Cisco Nexus 3000 Series switches, such as C3132Q-V , N3K-C31108PC-V and N3K-C31108TC-V.

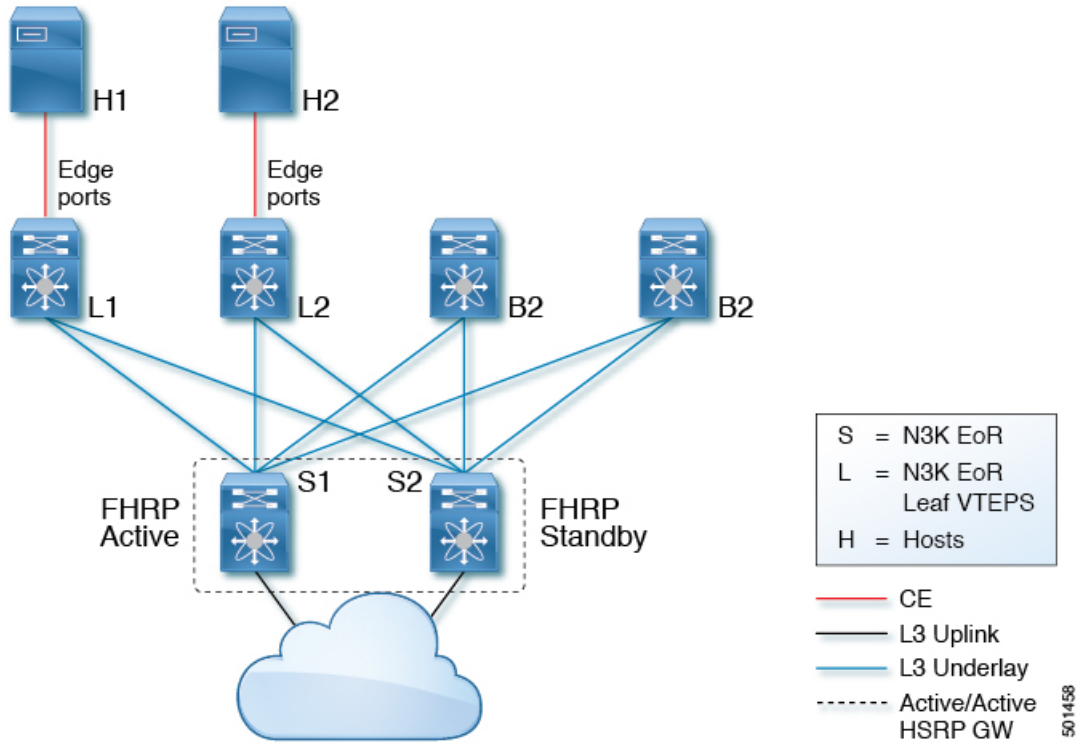
FHRP Over VXLAN Topology

In the following topology, the FHRP is configured on the Spine Layer. The FHRP protocols synchronize its state with the hellos that get flooded on the overlay without having a dedicated Layer 2 link in between the peers. The FHRP operates in an active/standby state as no vPC is deployed.

Note Bi-Directional Forwarding (BFD) is not supported with HSRP in the new topology.

The following image illustrates the new topology that supports a FHRP over VXLAN configuration:

Figure 7: Configuring FHRP Over VXLAN



Following is the configuration example of the new topology:

```
S1 FHRP configuration with HSRP
# VLAN with VNI
vlan 10
  vn-segment 10000

# Layer-3 Interface with FHRP (HSRP)
interface vlan 10
 ip address 192.168.1.2
  hsrp 10
  ip 192.168.1.1
```

```
S2 FHRP configuration with HSRP
# VLAN with VNI
vlan 10
  vn-segment 10000

# Layer-3 Interface with FHRP (HSRP)
interface vlan 10
 ip address 192.168.1.3
  hsrp 10
  ip 192.168.1.1
```

Note The FHRP configuration can leverage HSRP or VRRP. No vPC peer-link is necessary and therefore no VLAN is allowed on the vPC peer-link. The VNI mapped to the VLAN must be configured on the NVE interface and it is associated with the used BUM replication mode (Multicast or Ingress Replication).

Considerations for VXLAN Deployment

The following are some of the considerations while deploying VXLANs:

- A loopback interface IP is used to uniquely identify a VTEP device in the transport network.
- To establish IP multicast routing in the core, an IP multicast configuration, PIM configuration, and Rendezvous Point (RP) configuration are required.
- You can configure VTEP-to-VTEP unicast reachability through any IGP protocol.
- You can configure a VXLAN UDP destination port as required. The default port is 4789.
- The default gateway for VXLAN VLANs should be provisioned on a different upstream router.
- VXLAN multicast traffic should always use the RPT shared tree.
- An RP for the multicast group on the VTEP is a supported configuration. However, you must configure the RP for the multicast group at the spine layer/upstream device. Because all multicast traffic traverses the RP, it is more efficient to have this traffic directed to a spine layer/upstream device.

vPC Guidelines and Limitations for VXLAN Deployment

- Starting with Release 7.0(3)I2(1), The VXLAN multicast encapsulation path has duplicate members of the VPC peer-link on the VPC peers. This design has been adopted to support anycast RP and the service orphan traffic. For all the access side traffic, now two copies of a packet are sent over the VPC peer-link on the multicast path, one native and one VXLAN header encapsulated.
- You must bind NVE to a loopback address that is separate form other loopback addresses required by Layer 3 protocols. Use a dedicated loopback address for VXLAN.
- Multicast traffic on a vPC that is hashed toward the non-DF switch traverses the multichassis EtherChannel trunk (MCT) and is encapsulated on the DF node.
- In a VXLAN vPC, consistency checks are performed to ensure that NVE configurations and VN-Segment configurations are identical across vPC peers.

- The router ID for unicast routing protocols must be different from the loopback IP address used for VTEP.
- Configure an SVI between vPC peers and advertise routes between the vPC peers by using a routing protocol with higher routing metric. This action ensures that the IP connectivity of the vPC node does not go down if one vPC node fails.

Configuration Guidelines for VXLAN VPC Setup and Expected Behaviors in Various Scenarios

- VPC peers must have identical configurations:
  - Consistent VLAN to VN-segment mapping.
  - Consistent NVE1 binding to the same loopback interface.
  - Using the same secondary IP address.
  - Using different primary IP addresses.
  - Consistent VNI to group mapping.
- For multicast, the VPC node that receives the (S, G) join from the RP (rendezvous point) becomes the DF (Designated Forwarder). On the DF node, the encapsulation routes are installed for multicast.
- The decap routes are installed based on the election of a decapper from between the VPC primary node and the VPC secondary node. The winner of the decap election is the node with the least cost to the RP.
- However, if the cost to the RP is the same for both nodes, the VPC primary node is elected. The winner of the decap election has the decap mroute installed. The other node does not have a decap route installed.
- On a VPC device, the BUM traffic (broadcast, unknown-unicast, and multicast traffic) from hosts is replicated on the the peer-link. A copy is made of every native packet and each native packet is sent across the peer-link to service the orphan-ports connected to the peer VPC switch.
- To prevent traffic loops in VXLAN networks, native packets ingressing the peer-link cannot be sent to an uplink. However, if the peer switch is the encapper, the copied packet traverses the peer-link and it is sent to the uplink.
- When the peer-link is shut, the loopback address on the VPC secondary is brought down and the status is Admin Shut. This is done so that the route to the loopback is withdrawn on the upstream and that the upstream can divert all the traffic to the VPC primary.

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

- When the peer-link is no-shut, the NVE loopback address is brought up again and the route is advertised upstream attracting the traffic.
- For VPC,
  - The loopback interface has 2 IP addresses: the primary IP address and the secondary IP address.
  - The primary IP address is unique and is used by Layer 3 protocols.

- The secondary IP address on loopback is necessary because the interface NVE uses it for the VTEP IP address.
- The secondary IP address must be same on both vPC peers.
- The VPC peer-gateway feature must be enabled on both peers.

- As a best practice, use peer-switch, peer gateway, ip arp sync, ipv6 nd sync configurations for improved convergence in VPC topologies.
- When the NVE or loopback is shut in VPC configurations:
  - If the NVE or loopback is shut only on the primary VPC switch, the global VxLAN VPC consistency checker fails. Then the NVE, loopback, and VPCs are taken down on the secondary VPC switch.
  - If the NVE or loopback is shut only on the secondary VPC switch, the global VXLAN VPC consistency checker fails. Then the NVE, loopback, and secondary VPC are brought down on the secondary. The traffic continues to flow through the primary VPC switch.
- As a best practice, you should keep both the NVE and loopback up on both the primary and secondary VPC switches.
- Redundant anycast RPs configured in the network for multicast load-balancing and RP redundancy are supported on VPC VTEP topologies.
- Enabling vpc peer-switch configuration is mandatory. For peer-switch functionality, at least one SVI is required to be enabled across the peer-link and also configured with PIM. This provides a backup path in the case when VTEP loses complete connectivity to the spine. Remote peer reachability is re-routed over the peer-link in this case.

Configuring VXLAN Traffic Forwarding

There are two options for forwarding broadcast, unknown unicast and multicast traffic on a VXLAN Layer 2 gateway. Layer 2 Mechanisms for Broadcast, Unknown Unicast, and Multicast Traffic, on page 115 provides more information about these two options.

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

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

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

Enabling and Configuring the PIM Feature

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

This is a prerequisite only for multicast replication.

Before you begin

Ensure that you have installed the LAN Base Services license.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **feature pim**	Enables PIM. By default, PIM is disabled.
Step 3	(Optional) switch(config)# **show running-config pim**	Shows the running-configuration information for PIM, including the **feature** command.
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 enable the PIM feature:

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

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

version 6.0(2)U3(1)
feature pim

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

Configuring a Rendezvous Point

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

This is a prerequisite only for multicast replication.

Before you begin

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

Procedure

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

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

Enabling a VXLAN

Enabling VXLANs involves the following:

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

Before you begin

Ensure that you have installed the VXLAN Enterprise license.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **[no] feature nv overlay**	Enables the VXLAN feature.
Step 3	switch (config)# **[no] feature vn-segment-vlan-based**	Configures the global mode for all VXLAN bridge domains.

	Command or Action	Purpose
		Enables VLAN to VN-Segment mapping. VLAN to VN-Segment mapping is always one-to-one.
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 enable a VXLAN and configure VLAN to VN-Segment mapping:

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

Mapping a VLAN to a VXLAN VNI

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **vlan** *vlan-id*	Specifies a VLAN.
Step 3	switch(config-vlan)# **vn-segment** *vnid*	Specifies the VXLAN virtual network identifier (VNID). The range of values for vnid is 1 to 16777214.

Example

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

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

Configuring a Routing Protocol for NVE Unicast Addresses

Configuring a routing protocol for unicast addresses involves the following:

- Configuring a dedicated loopback interface for NVE reachability.
- Configuring the routing protocol network type.
- Specifying the routing protocol instance and area for an interface.

- Enabling PIM sparse mode in case of multicast replication.

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

This is a prerequisite for both multicast and ingress replication.

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

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

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface loopback** *instance*	Creates a dedicated loopback interface for the NVE interface. The instance range is from 0 to 1023.
Step 3	switch(config-if)# **ip address** *ip-address/length*	Configures an IP address for this interface.
Step 4	switch(config-if)# **ip ospf network** {**broadcast** | **point-to-point**}	Configures the OSPF network type to a type other than the default for an interface.
Step 5	switch(config-if)# **ip router ospf** *instance-tag* **area** *area-id*	Specifies the OSPF instance and area for an interface.
Step 6	switch(config-if)# **ip pim sparse-mode**	Enables PIM sparse mode on this interface. The default is disabled. Enable the PIM sparse mode in case of multicast replication.

Example

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

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

Creating a VXLAN Destination UDP Port

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

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

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

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

Procedure

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

Example

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

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

Creating and Configuring an NVE Interface

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

Procedure

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

Example

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

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

Configuring Replication for a VNI

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

- Multicast replication
- Ingress replication

Configuring Multicast Replication

Before you begin

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

Procedure

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

Example

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

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

Configuring Ingress Replication

Before you begin

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

Procedure

	Command or Action	Purpose
Step 1	switch(config-if-nve)# **member vni** *vnid*	Maps VXLAN VNIs to the NVE interface.

	Command or Action	Purpose
Step 2	switch(config-if-nve-vni)# **ingress-replication protocol static**	Enables static ingress replication for the VNI.
Step 3	switch(config-if-nve-vni)# **peer-ip** *ip-address*	Enables the peer IP. **Note** • A VNI can be associated only with a single IP address. • An IP address can be associated only with a single VNI.

Example

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

```
switch(config-if-nve)# member vni 5001
switch(config-if-nve-vni)# ingress-replication protocol static
switch(config-if-nve-vni)# peer-ip 111.1.1.1
```

Configuring Q-in-VNI

Using Q-in-VNI provides a way for you to segregate traffic by mapping to a specific port. In a multi-tenant environment, you can specify a port to a tenant and send/receive packets over the VXLAN overlay.

Notes about configuring a Q-in-VNI:

- Q-in-VNI is supported only for the Cisco Nexus 3100-V and 3132C-Z platform switches.
- The dot1q mode is not supported for 40G ports.
- Beginning with Cisco NX-OS 7.0(3)I5(1), Q-in-Q to Q-in-VNI interworking is supported.
- Q-in-VNI only supports VXLAN bridging. It does not support VXLAN routing.
- Q-in-VNI does not support FEX.
- When configuring access ports and trunk ports:
  - For Cisco NX-OS 7.0(3)I2(2) and earlier releases, when a switch is in dot1q mode, you cannot have access ports or trunk ports configured on any other interface on the switch.
  - For Cisco NX-OS 7.0(3)I3(1) and later releases, you can have access ports, trunk ports and dot1q ports on different interfaces on the same switch.
  - For Cisco NX-OS 7.0(3)I3(1) and later releases, you cannot have the same VLAN configured for both dot1q and trunk ports/access ports.

Before you begin

Configuring the Q-in-VNI feature requires:

- The base port mode must be a dot1q tunnel port with an access VLAN configured.
- VNI mapping is required for the access VLAN on the port.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**interface** *type port*	Enters interface configuration mode.
Step 3	**switchport mode dot1q-tunnel**	Creates a 802.1Q tunnel on the port.
Step 4	**switchport access vlan** *vlan-id*	Specifies the port assigned to a VLAN.
Step 5	**spanning-tree bpdufilter enable**	Enables BPDU Filtering for the specified spanning tree edge interface. By default, BPDU Filtering is disabled.
Step 6	**interface nve** *x*	Creates a VXLAN overlay interface that terminates VXLAN tunnels. **Note** This step is required for Cisco NX-OS 7.0(3)I2(2) and earlier releases. This step is not required for Cisco NX-OS 7.0(3)I3(1) and later releases.
Step 7	**overlay-encapsulation vxlan-with-tag**	Enables Q-in-VNI. **Note** This step is required for Cisco NX-OS 7.0(3)I2(2) and earlier releases: This step is not required for Cisco NX-OS 7.0(3)I3(1) and later releases.

Example

- The following example shows how to configure Q-in-VNI (Cisco NX-OS 7.0(3)I2(2) and earlier releases):

```
switch# config terminal
switch(config)# interface ethernet 1/4
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# switchport access vlan 10
switch(config-if)# spanning-tree bpdufilter enable
switch(config-if)# interface nve1
switch(config-if)# overlay-encapsulation vxlan-with-tag
```

- The following example shows how to configure Q-in-VNI (Cisco NX-OS 7.0(3)I3(1) and later releases):

```
switch# config terminal
switch(config)# interface ethernet 1/4
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# switchport access vlan 10
switch(config-if)# spanning-tree bpdufilter enable
switch(config-if)#
```

Verifying the VXLAN Configuration

Use one of the following commands to verify the VXLAN configuration, to display the MAC addresses, and to clear the MAC addresses:

Command	Purpose
show nve interface nve *id*	Displays the configuration of an NVE interface.
show nve vni	Displays the VNI that is mapped to an NVE interface.
show nve peers	Displays peers of the NVE interface.
show interface nve *id* **counters**	Displays all the counters for an NVE interface.
show nve vxlan-params	Displays the VXLAN UDP port configured.
show mac address-table	Displays both VLAN and VXLAN MAC addresses.
clear mac address-table dynamic	Clears all MAC address entries in the MAC address table.

Example

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

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

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

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

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

```
switch# show nve vni
Interface          VNI        Multicast-group   VNI State
-----------------  --------   ---------------   ---------
```

```
nve1               5000       0.0.0.0           Up
```

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

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

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

```
switch# show interface nv 1 counter

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

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

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

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

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

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

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

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

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

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

Overview of IGMP Snooping Over VXLAN

The configuration of IGMP snooping is same in VXLAN as in configuration of IGMP snooping in regular VLAN domain. All the configuration CLIs remain the same. For more information on IGMP snooping, see the *Configuring IGMP Snooping* section in *Cisco Nexus 3000 Series NX-OS Multicast Routing Configuration Guide, Release 7.x*.

Guidelines and Limitations for IGMP Snooping Over VXLAN

See the following guidelines and limitations for IGMP snooping over VXLAN:

- Starting with Cisco NX-OS Release 7.0(3)I5(1), IGMP snooping over VXLAN is supported.
- IGMP snooping on VXLAN VLAN is disabled by default.
- For IGMP snooping over VXLAN, all the guidelines and limitations of VXLAN apply.
- IGMP snooping over VXLAN is not supported on any FEX enabled platforms and FEX ports.
- IGMP snooping over VXLAN VLAN is supported on Cisco Nexus 3132Q (N9K mode only), 3172 (N9K mode only), and 3100-V platform switches.

Configuring IGMP Snooping Over VXLAN

Before you begin

For VXLAN IGMP snooping functionality, the ARP-ETHER TCAM must be configured in the double-wide mode using the CLI command, switch# **hardware access-list tcam region arp-ether 256 double wide**.

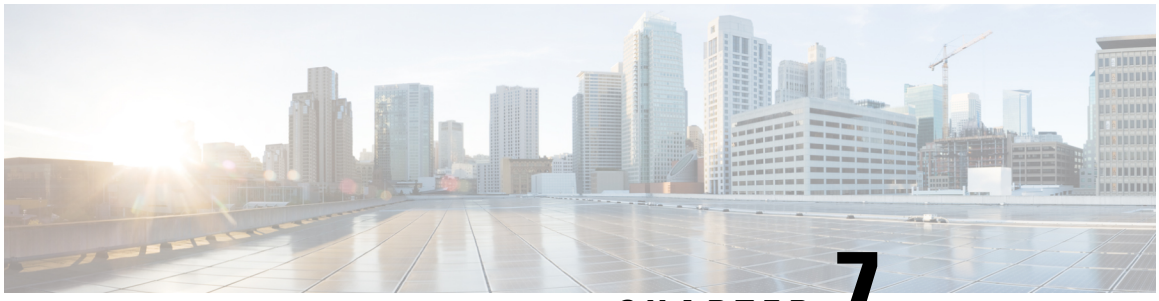
Procedure

	Command or Action	Purpose
Step 1	switch(config)#**ip igmp snooping vxlan**	Enables IGMP snooping for VXLAN VLANs. You have to explicitly configure this command to enable snooping for VXLAN VLANs.
Step 2	switch(config)#**ip igmp snooping disable-nve-static-router-port**	Configures IGMP snooping over VXLAN to not include NVE as static mrouter port using this global CLI command. IGMP snooping over VXLAN has the NVE interface as mrouter port by default.

	Command or Action	Purpose

Step 3	switch(config)#**system nve ipmc global index-size ?**
	Example:
	`switch(config)# system nve ipmc global index-size ?`
	`<1000-7000> Ipmc allowed size`
		Configures the VXLAN global IPMC index size. IGMP snooping over VXLAN uses the IPMC indexes from the NVE global range on the Cisco Nexus 3000 Series switches with Network Forwarding Engine (NFE). You need to reconfigure the VXLAN global IPMC index size according to the scale using this command. Cisco recommends to reserve 6000 IPMC indexes using this CLI command. The default IPMC index size is 3000.

Step 4	switch(config)# **ip igmp snooping vxlan-umc drop vlan ?**
	Example:
	`switch(config)# ip igmp snooping vxlan-umc drop vlan ?`
	`<1-3863>  VLAN IDs for which unknown multicast traffic is dropped`
		Configures IGMP snooping over VXLAN to drop all the unknown multicast traffic on per VLAN basis using this global CLI command. On Cisco Nexus 3000 Series switches with Network Forwarding Engine (NFE), the default behavior of all unknown multicast traffic is to flood to the bridge domain.



CHAPTER 7

Configuring VXLAN BGP EVPN

This chapter contains the following sections:

- Information About VXLAN BGP EVPN, on page 135
- Configuring VXLAN BGP EVPN, on page 145
- Verifying the VXLAN BGP EVPN Configuration, on page 156
- Example of VXLAN BGP EVPN (EBGP), on page 157
- Example of VXLAN BGP EVPN (IBGP), on page 166
- Example Show Commands, on page 175

Information About VXLAN BGP EVPN

Guidelines and Limitations for VXLAN BGP EVPN

VXLAN BGP EVPN has the following guidelines and limitations:

- The following guidelines and limitations apply to VXLAN/VTEP:
  - SPAN source or destination is supported on any port.

  For more information, see the Cisco Nexus 9000 Series NX-OS System Management Configuration Guide, Release 7.x.
- When SVI is enabled on a VTEP (flood and learn, or EVPN) regardless of ARP suppression, make sure that ARP-ETHER TCAM is carved using the **hardware access-list tcam region arp-ether 256 double-wide** command. This is not applicable to the Cisco Nexus 9200 and 9300-EX platform switches and Cisco Nexus 9500 platform switches with 9700-EX line cards.
- Beginning with Cisco NX-OS Release 7.0(3)F3(3), VXLAN Layer 2 Gateway is supported only on the 9636C-RX line card. VXLAN and MPLS cannot be enabled on the Cisco Nexus 9508 switch at the same time.
- Beginning with Cisco NX-OS Release 7.0(3)F3(3), if VXLAN is enabled, the Layer 2 Gateway cannot be enabled when there is any line card other than the 9636C-RX.
- Beginning with Cisco NX-OS Release 7.0(3)I6(1), you can configure EVPN over segment routing or MPLS. See the Cisco Nexus 9000 Series NX-OS Label Switching Configuration Guide, Release 7.x for more information.

- Beginning with Cisco NX-OS Release 7.0(3)I6(1), you can use MPLS tunnel encapsulation using the new CLI encapsulation mpls command. You can configure the label allocation mode for the EVPN address family. See the Cisco Nexus 9000 Series NX-OS Label Switching Configuration Guide, Release 7.x for more information.
- In VXLAN EVPN setup that has 2K VNI scale configuration, the control plane down time takes more than 200 seconds. To avoid BGP flap, configure the graceful restart time to 300 seconds.
- SVI and subinterfaces as uplinks are not supported.
- In a VXLAN EVPN setup, border leaves must use unique route distinguishers, preferably using **auto rd** command. It is not supported to have same route distinguishers in different border leaves.
- ARP suppression is only supported for a VNI if the VTEP hosts the First-Hop Gateway (Distributed Anycast Gateway) for this VNI. The VTEP and the SVI for this VLAN have to be properly configured for the distributed Anycast Gateway operation, for example, global Anycast Gateway MAC address configured and Anycast Gateway feature with the virtual IP address on the SVI.
- When Layer 3 EVPN is configured in Cisco Nexus 3000 Series switches that are based on Broadcom ASIC and these switches are added in the topology with Layer 2 EVPN, the routing for this scenario is not supported. When you configure SVI and Layer 3 EVPN in Cisco Nexus 3000 Series switches that are based on a Broadcom ASIC with Anycast Gateway and when you send the ARP requests from a Layer 2 EVPN device (for example, Cisco Nexus 3000 Series switches that are based on a Broadcom ASIC), the Cisco Nexus 3000 Series switches cannot be used as a gateway for the ARP requests received on the network ports.
- The **show** commands with the **internal** keyword are not supported.
- DHCP snooping (Dynamic Host Configuration Protocol snooping) is not supported on VXLAN VLANs.
- RACLs are not supported on Layer 3 uplinks for VXLAN traffic. Egress VACLs support is not available for de-capsulated packets in the network to access direction on the inner payload.

  As a best practice, use PACLs/VACLs for the access to the network direction.
- QoS classification is not supported for VXLAN traffic in the network to access direction on the Layer 3 uplink interface.
- The QoS buffer-boost feature is not applicable for VXLAN traffic.
- VTEP does not support Layer 3 subinterface uplinks that carry VXLAN encapsulated traffic.
- Layer 3 interface uplinks that carry VXLAN encapsulated traffic do not support subinterfaces for non-VXLAN encapsulated traffic.
- On Cisco Nexus 9000 PX/TX/PQ switches configured as VXLAN VTEPs, if any ALE 40G port is used as a VXLAN underlay port, configuring subinterfaces on either this or any other 40G port is not allowed and could lead to VXLAN traffic loss.
- For eBGP, it is recommended to use a single overlay eBGP EVPN session between loopbacks.
- EBGP peering from a VXLAN host to local VTEP is supported with loopback in tenant VRF as BGP update-source.
- You must bind NVE to a loopback address that is separate from other loopback addresses that are required by Layer 3 protocols. NVE and other Layer 3 protocols using the same loopback is not supported.
- VXLAN BGP EVPN does not support an NVE interface in a non-default VRF.

- It is recommended to configure a single BGP session over the loopback for an overlay BGP session.
- When configuring VXLAN BGP EVPN, only the "System Routing Mode: Default" is applicable for the following hardware platforms:
  - Cisco Nexus 9200, 9300-EX, and 9300-FX/FX2 platform switches
  - Cisco Nexus 9300 platform switches
  - Cisco Nexus 9500 platform switches with X9500 line cards
  - Cisco Nexus 9500 platform switches with -EX and -FX line cards
- The “System Routing Mode: template-vxlan-scale” is not applicable to Cisco NX-OS Release 7.0(3)I5(2) and later.
- When using VXLAN BGP EVPN with Cisco NX-OS Release 7.0(3)I4(x) or 7.0(3)I5(1), the “System Routing Mode: template-vxlan-scale” is required on the following hardware platforms:
  - Cisco Nexus 9300-EX platform switches
  - Cisco Nexus 9500 platform switches with -EX line cards
- Changing the “System Routing Mode” requires a reload of the switch.
- The VXLAN UDP port number is used for VXLAN encapsulation. For Cisco Nexus NX-OS, the UDP port number is 4789. It complies with IETF standards and is not configurable.
- For Cisco NX-OS Release 7.0(3)I4(1) and later, VXLAN supports In-Service Software Upgrade (ISSU).
- The **vpc orphan-ports suspend** command must be enabled for orphan ports that are connected to Cisco Nexus 9000 vPC VTEPs.
- VTEP connected to FEX host interface ports is not supported (7.0(3)I2(1) and later).
- In Cisco NX-OS Release 7.0(3)I4(1), 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.

Note For information about VXLAN BGP EVPN scalability, see the Verified Scalability Guide for your platform.

Notes for EVPN Convergence

The following are notes about EVPN Convergence (7.0(3)I3(1) and later):

- As a best practice, the NVE source loopback should be dedicated to NVE. so that NVE can bring the loopback up or down as needed.
- When vPC has been configured, the loopback stays down until the MCT link comes up.

Note When **feature vpc** is enabled and there is no VPC configured, the NVE source loopback is in "shutdown" state after an upgrade. In this case, removing **feature vpc** restores the interface to "up" state.

- The NVE underlay (through the source loopback) is kept down until the overlay has converged.
  - When MCT comes up, the source loopback is kept down for an amount of time that is configurable. This approach prevents north-south traffic from coming in until the overlay has converged.
  - When MCT goes down, NVE is kept up for 30 seconds in the event that there is still south-north traffic from vPC legs which have not yet gone down.
- BGP ignores routes from vPC peer. This reduces the number of routes in BGP.

Considerations for VXLAN BGP EVPN Deployment

- A loopback address is required when using the **source-interface config** command. The loopback address represents the local VTEP IP.
- During boot-up of a switch (7.0(3)I2(2) and later), you can use the **source-interface hold-down-time** *hold-down-time* command to suppress advertisement of the NVE loopback address until the overlay has converged. The range for the *hold-down-time* is 0 - 1000 seconds. The default is 180 seconds.
- To establish IP multicast routing in the core, IP multicast configuration, PIM configuration, and RP configuration is required.
- VTEP to VTEP unicast reachability can be configured through any IGP/BGP protocol.
- If the anycast gateway feature is enabled for a specific VNI, then the anyway gateway feature must be enabled on all VTEPs that have that VNI configured. Having the anycast gateway feature configured on only some of the VTEPs enabled for a specific VNI is not supported.
- It is a requirement when changing the primary or secondary IP address of the NVE source interfaces to shut the NVE interface before changing the IP address.
- As a best practice, the RP for the multicast group should be configured only on the spine layer. Use the anycast RP for RP load balancing and redundancy.
- Every tenant VRF needs a VRF overlay VLAN and SVI for VXLAN routing.
- For scale environments, the VLAN IDs related to the VRF and Layer-3 VNI (L3VNI) must be reserved with the **system vlan nve-overlay id** command.

  The following example shows how to reserve the VLAN IDs related to the VRF and the Layer-3 VNI:

```
system vlan nve-overlay id 2000

vlan 2000
  vn-segment 50000

interface Vlan2000
  vrf member MYVRF_50000
  ip forward
  ipv6 forward
```

```
vrf context MYVRF_50000
  vni 50000
```

Note The **system vlan nve-overlay id** command should be used for a VRF or a Layer-3 VNI (L3VNI) only. Do not use this command for regular VLANs or Layer-2 VNIs (L2VNI).

- When configuring ARP suppression with BGP-EVPN, use the **hardware access-list tcam region arp-ether** *size* **double-wide** command to accommodate ARP in this region. (You must decrease the size of an existing TCAM region before using this command.)

vPC Considerations for VXLAN BGP EVPN Deployment

- The loopback address used by NVE needs to be configured to have a primary IP address and a secondary IP address.

  The secondary IP address is used for all VxLAN traffic that includes multicast and unicast encapsulated traffic.

- Each vPC peer needs to have separate BGP sessions to the spine.
- vPC peers must have identical configurations.
  - Consistent VLAN to VN-segment mapping.
  - Consistent NVE1 binding to the same loopback interface
    - Using the same secondary IP address.
    - Using different primary IP addresses.
  - Consistent VNI to group mapping.
  - The VRF overlay VLAN should be a member of the peer-link port-channel.
- For multicast, the vPC node that receives the (S, G) join from the RP (rendezvous point) becomes the DF (designated forwarder). On the DF node, encap routes are installed for multicast.

  Decap routes are installed based on the election of a decapper from between the vPC primary node and the vPC secondary node. The winner of the decap election is the node with the least cost to the RP. However, if the cost to the RP is the same for both nodes, the vPC primary node is elected.

  The winner of the decap election has the decap mroute installed. The other node does not have a decap route installed.

- On a vPC device, BUM traffic (broadcast, unknown-unicast, and multicast traffic) from hosts is replicated on the peer-link. A copy is made of every native packet and each native packet is sent across the peer-link to service orphan-ports connected to the peer vPC switch.

  To prevent traffic loops in VXLAN networks, native packets ingressing the peer-link cannot be sent to an uplink. However, if the peer switch is the encapper, the copied packet traverses the peer-link and is sent to the uplink.

Note Each copied packet is sent on a special internal VLAN (VLAN 4041).

- When peer-link is shut, the loopback interface used by NVE on the vPC secondary is brought down and the status is **Admin Shut**. This is done so that the route to the loopback is withdrawn on the upstream and that the upstream can divert all traffic to the vPC primary.

Note Orphans connected to the vPC secondary will experience loss of traffic for the period that the peer-link is shut. This is similar to Layer 2 orphans in a vPC secondary of a traditional vPC setup.

- When the vPC domain is shut, the loopback interface used by NVE on the VTEP with shutdown vPC domain is brought down and the status is **Admin Shut**. This is done so that the route to the loopback is withdrawn on the upstream and that the upstream can divert all traffic to the other vPC VTEP.
- When peer-link is no-shut, the NVE loopback address is brought up again and the route is advertised upstream, attracting traffic.
- For vPC, the loopback interface has 2 IP addresses: the primary IP address and the secondary IP address.

  The primary IP address is unique and is used by Layer 3 protocols.

  The secondary IP address on loopback is necessary because the interface NVE uses it for the VTEP IP address. The secondary IP address must be same on both vPC peers.
- The vPC peer-gateway feature must be enabled on both peers.

  As a best practice, use peer-switch, peer gateway, ip arp sync, ipv6 nd sync configurations for improved convergence in vPC topologies.

  In addition, increase the STP hello timer to 4 seconds to avoid unnecessary TCN generations when vPC role changes occur.

  The following is an example (best practice) of a vPC configuration:

```
switch# sh ru vpc

version 6.1(2)I3(1)
feature vpc
vpc domain 2
  peer-switch
  peer-keepalive destination 172.29.206.65 source 172.29.206.64
  peer-gateway
  ipv6 nd synchronize
  ip arp synchronize
```

- On a vPC pair, shutting down NVE or NVE loopback on one of the vPC nodes is not a supported configuration. This means that traffic failover on one-side NVE shut or one-side loopback shut is not supported.
- Redundant anycast RPs configured in the network for multicast load-balancing and RP redundancy are supported on vPC VTEP topologies.

- Enabling vpc peer-gateway configuration is mandatory. For peer-gateway functionality, at least one backup routing SVI is required to be enabled across peer-link and also configured with PIM. This provides a backup routing path in the case when VTEP loses complete connectivity to the spine. Remote peer reachability is re-routed over the peer-link in this case.

  The following is an example of SVI with PIM enabled:

```
switchh# sh ru int vlan 2

interface Vlan2
  description special_svi_over_peer-link
  no shutdown
  ip address 30.2.1.1/30
  ip pim sparse-mode
```

Note The SVI must be configured on both vPC peers and requires PIM to be enabled.

- As a best practice when changing the secondary IP address of an anycast vPC VTEP, the NVE interfaces on both the vPC primary and the vPC secondary should be shut before the IP changes are made.
- To provide redundancy and failover of VXLAN traffic when a VTEP loses all of its uplinks to the spine, it is recommended to run a Layer 3 link or an SVI link over the peer-link between vPC peers.
- If DHCP Relay is required in VRF for DHCP clients or if loopback in VRF is required for reachability test on a vPC pair, it is necessary to create a backup SVI per VRF with PIM enabled.

```
switchh# sh ru int vlan 20

interface Vlan20
description backup routing svi for VRF Green
vrf member GREEN
no shutdown
ip address 30.2.10.1/30
```

Network Considerations for VXLAN Deployments

- MTU Size in the Transport Network

  Due to the MAC-to-UDP encapsulation, VXLAN introduces 50-byte overhead to the original frames. Therefore, the maximum transmission unit (MTU) in the transport network needs to be increased by 50 bytes. If the overlays use a 1500-byte MTU, the transport network needs to be configured to accommodate 1550-byte packets at a minimum. Jumbo-frame support in the transport network is required if the overlay applications tend to use larger frame sizes than 1500 bytes.

- ECMP and LACP Hashing Algorithms in the Transport Network

  As described in a previous section, Cisco Nexus 3000 Series Switches introduce a level of entropy in the source UDP port for ECMP and LACP hashing in the transport network. As a way to augment this implementation, the transport network uses an ECMP or LACP hashing algorithm that takes the UDP source port as an input for hashing, which achieves the best load-sharing results for VXLAN encapsulated traffic.

- Multicast Group Scaling

The VXLAN implementation on Cisco Nexus 3000 Series Switches uses multicast tunnels for broadcast, unknown unicast, and multicast traffic forwarding. Ideally, one VXLAN segment mapping to one IP multicast group is the way to provide the optimal multicast forwarding. It is possible, however, to have multiple VXLAN segments share a single IP multicast group in the core network. VXLAN can support up to 16 million logical Layer 2 segments, using the 24-bit VNID field in the header. With one-to-one mapping between VXLAN segments and IP multicast groups, an increase in the number of VXLAN segments causes a parallel increase in the required multicast address space and the amount of forwarding states on the core network devices. At some point, multicast scalability in the transport network can become a concern. In this case, mapping multiple VXLAN segments to a single multicast group can help conserve multicast control plane resources on the core devices and achieve the desired VXLAN scalability. However, this mapping comes at the cost of suboptimal multicast forwarding. Packets forwarded to the multicast group for one tenant are now sent to the VTEPs of other tenants that are sharing the same multicast group. This causes inefficient utilization of multicast data plane resources. Therefore, this solution is a trade-off between control plane scalability and data plane efficiency.

Despite the suboptimal multicast replication and forwarding, having multiple-tenant VXLAN networks to share a multicast group does not bring any implications to the Layer 2 isolation between the tenant networks. After receiving an encapsulated packet from the multicast group, a VTEP checks and validates the VNID in the VXLAN header of the packet. The VTEP discards the packet if the VNID is unknown to it. Only when the VNID matches one of the VTEP's local VXLAN VNIDs, does it forward the packet to that VXLAN segment. Other tenant networks will not receive the packet. Thus, the segregation between VXLAN segments is not compromised.

Considerations for the Transport Network

The following are considerations for the configuration of the transport network:

- On the VTEP device:
  - Enable and configure IP multicast.*
  - Create and configure a loopback interface with a /32 IP address.

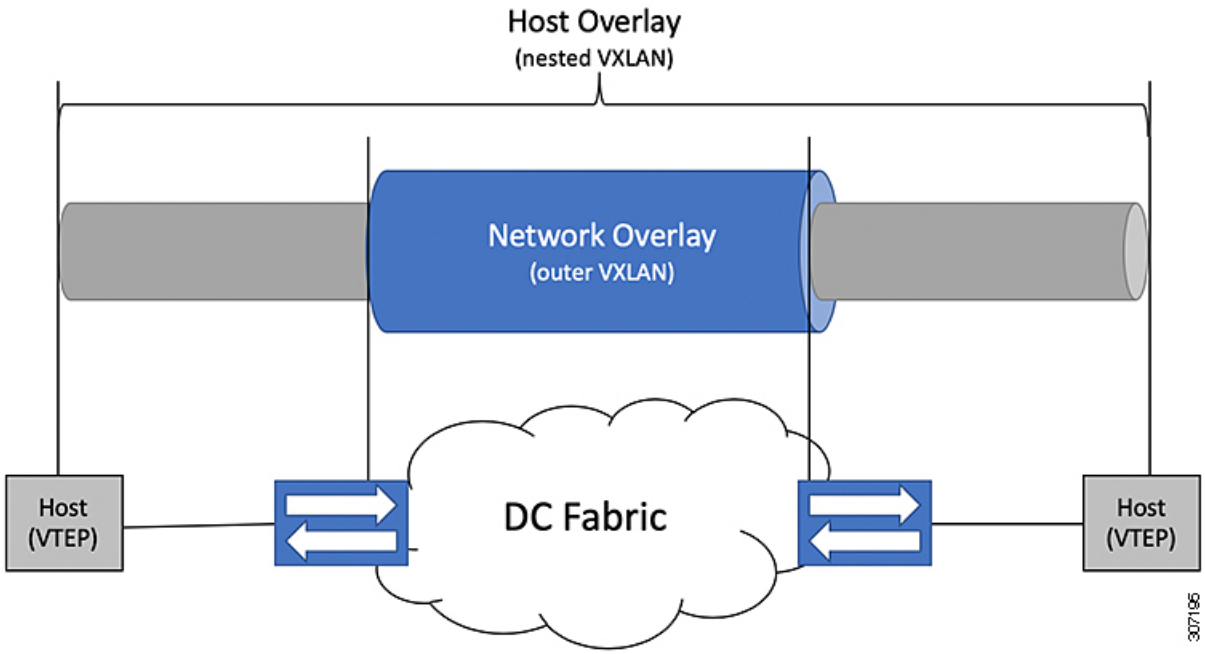
    (For vPC VTEPs, you must configure primary and secondary /32 IP addresses.)
  - Enable UP multicast on the loopback interface. *
  - Advertise the loopback interface /32 addresses throught the routing protocol (static route) that runs in the transport network.
  - Enable IP multicast on the uplink outgoing physical interface. *
- Throughout the transport network:
  - Enable and configure IP multicast.*

Note * Not required for static ingress replication or BGP EVPN ingress replication.

Considerations for Tunneling VXLAN

DC Fabrics with VXLAN BGP EVPN are becoming the transport infrastructure for overlays. These overlays, often originated on the server (Host Overlay), require integration or transport over the top of the existing transport infrastructure (Network Overlay).

Figure 8: Host Overlay



To provide Nested VXLAN support, the switch hardware and software must differentiate between two different VXLAN profiles:

- VXLAN originated behind the Hardware VTEP for transport over VXLAN BGP EVPN (nested VXLAN)
- VXLAN originated behind the Hardware VTEP to integrated with VXLAN BGP EVPN (BUD Node)

The detection of the two different VXLAN profiles is automatic and no specific configuration is needed for nested VXLAN. As soon as VXLAN encapsulated traffic arrives in a VXLAN enabled VLAN, the traffic is transported over the VXLAN BGP EVPN enabled DC Fabric.

The following attachment modes are supported for Nested VXLAN:

- Untagged traffic (in native VLAN on a trunk port or on an access port)
- Tagged traffic (tagged VLAN on a IEEE 802.1Q trunk port)
- Untagged and tagged traffic that is attached to a vPC domain
- Untagged traffic on a Layer 3 interface of a Layer 3 port-channel interface

BGP EVPN Considerations for VXLAN Deployment

Commands for BGP EVPN

The following describes commands to support BGP EVPN VXLAN control planes.

Command	Description
member vni *range* [**associate-vrf**]	Associate VXLAN VNIs (Virtual Network Identifiers) with the NVE interface The attribute **associate- vrf** is used to identify and separate processing VNIs that are associated with a VRF and used for routing. **Note** The VRF and VNI specified with this command must match the configuration of the VNI under the VRF.
show nve vni **show nve vni summary**	Displays information that determine if the VNI is configured for peer and host learning via the control plane or data plane.
show bgp l2vpn evpn **show bgp l2vpn evpn summary**	Displays the Layer 2 VPN EVPN address family.
host-reachability protocol bgp	Specifies BGP as the mechanism for host reachability advertisement.
suppress-arp	Suppresses ARP under Layer 2 VNI.
fabric forwarding anycast-gateway-mac	Configures anycast gateway MAC of the switch.
vrf context	Creates the VRF and enter the VRF mode.
nv overlay evpn	Enables/Disables the Ethernet VPN (EVPN).
router bgp	Configures the Border Gateway Protocol (BGP).

Command	Description
suppress mac-route	Suppresses the BGP MAC route so that BGP only sends the MAC/IP route for a host. Under NVE, the MAC updates for all VNIs are suppressed. **Note** • Receive-side — Suppressing the MAC route depends upon the capability of the remote EVPN peer to derive a MAC route from the MAC/IP route (7.0(3)I2(2) and later). Avoid using the "suppress mac-route" command if devices in the network are running an earlier NX-OS release. • Send-side — Suppressing the MAC route means that the sender has a MAC/IP route. If your configuration has pure-Layer 2 VNIs (such as no corresponding VRF or Layer3-VNI), then there is no corresponding MAC/IP and you should avoid using the "suppress mac-route" command.

Configuring VXLAN BGP EVPN

Enabling VXLAN

Enable VXLAN and the EVPN.

Procedure

	Command or Action	Purpose
Step 1	**feature vn-segment**	Enable VLAN-based VXLAN
Step 2	**feature nv overlay**	Enable VXLAN
Step 3	**nv overlay evpn**	Enable the EVPN control plane for VXLAN.

Configuring VLAN and VXLAN VNI

Procedure

	Command or Action	Purpose
Step 1	**vlan** *number*	Specify VLAN.
Step 2	**vn-segment** *number*	Map VLAN to VXLAN VNI to configure Layer 2 VNI under VXLAN VLAN.

Configuring VRF for VXLAN Routing

Configure the tenant VRF.

Procedure

	Command or Action	Purpose
Step 1	**vrf context** *vxlan*	Configure the VRF.
Step 2	**vni** *number*	Specify VNI.
Step 3	**rd auto**	Specify VRF RD (route distinguisher).
Step 4	**address-family ipv4 unicast**	Configure address family for IPv4.
Step 5	**route-target both auto**	**Note** Specifying the **auto** option is applicable only for IBGP. Manually configured route targets are required for EBGP.
Step 6	**route-target both auto evpn**	**Note** Specifying the **auto** option is applicable only for IBGP. The **auto** option is available beginning with Cisco NX-OS Release 7.0(3)I7(1). Manually configured route targets are required for EBGP.
Step 7	**address-family ipv6 unicast**	Configure address family for IPv6.
Step 8	**route-target both auto**	**Note** Specifying the **auto** option is applicable only for IBGP. The **auto** option is available beginning with Cisco NX-OS Release 7.0(3)I7(1). Manually configured route targets are required for EBGP.

	Command or Action	Purpose	
Step 9	**route-target both auto evpn**	**Note**	Specifying the **auto** option is applicable only for IBGP. Manually configured route targets are required for EBGP.

About RD Auto

The auto-derived Route Distinguisher (rd auto) is based on the Type 1 encoding format as described in IETF RFC 4364 section 4.2 https://tools.ietf.org/html/rfc4364#section-4.2. The Type 1 encoding allows a 4-byte administrative field and a 2-byte numbering field. Within Cisco NX-OS, the auto derived RD is constructed with the IP address of the BGP Router ID as the 4-byte administrative field (RID) and the internal VRF identifier for the 2-byte numbering field (VRF ID).

The 2-byte numbering field is always derived from the VRF, but results in a different numbering scheme depending on its use for the IP-VRF or the MAC-VRF:

- The 2-byte numbering field for the IP-VRF uses the internal VRF ID starting at 1 and increments. VRF IDs 1 and 2 are reserved for the default VRF and the management VRF respectively. The first custom defined IP VRF uses VRF ID 3.
- The 2-byte numbering field for the MAC-VRF uses the VLAN ID + 32767, which results in 32768 for VLAN ID 1 and incrementing.

Example auto-derived Route Distinguisher (RD)

- IP-VRF with BGP Router ID 192.0.2.1 and VRF ID 6 - RD 192.0.2.1:6
- MAC-VRF with BGP Router ID 192.0.2.1 and VLAN 20 - RD 192.0.2.1:32787

About Route-Target Auto

The auto-derived Route-Target (route-target import/export/both auto) is based on the Type 0 encoding format as described in IETF RFC 4364 section 4.2 (https://tools.ietf.org/html/rfc4364#section-4.2). IETF RFC 4364 section 4.2 describes the Route Distinguisher format and IETF RFC 4364 section 4.3.1 refers that it is desirable to use a similar format for the Route-Targets. The Type 0 encoding allows a 2-byte administrative field and a 4-byte numbering field. Within Cisco NX-OS, the auto derived Route-Target is constructed with the Autonomous System Number (ASN) as the 2-byte administrative field and the Service Identifier (VNI) for the 4-byte numbering field.

Examples of an auto derived Route-Target (RT):

- IP-VRF within ASN 65001 and L3VNI 50001 - Route-Target 65001:50001
- MAC-VRF within ASN 65001 and L2VNI 30001 - Route-Target 65001:30001

For Multi-AS environments, the Route-Targets must either be statically defined or rewritten to match the ASN portion of the Route-Targets.

Note Auto derived Route-Targets for a 4-byte ASN are not supported.

Configuring SVI for Hosts for VXLAN Routing

Configure the SVI for hosts.

Procedure

	Command or Action	Purpose
Step 1	**vlan** *number*	Specify VLAN
Step 2	**interface** *vlan-number*	Specify VLAN interface.
Step 3	**vrf member** *vxlan-number*	Configure SVI for host.
Step 4	**ip address** *address*	Specify IP address.

Configuring VRF Overlay VLAN for VXLAN Routing

Procedure

	Command or Action	Purpose
Step 1	**vlan** *number*	Specify VLAN.
Step 2	**vn-segment** *number*	Specify vn-segment.

Configuring VNI Under VRF for VXLAN Routing

Configures a Layer 3 VNI under a VRF overlay VLAN. (A VRF overlay VLAN is a VLAN that is not associated with any server facing ports. All VXLAN VNIs that are mapped to a VRF, need to have their own internal VLANs allocated to it.)

Procedure

	Command or Action	Purpose
Step 1	**vrf context** *vxlan*	Create a VXLAN Tenant VRF
Step 2	**vni** *number*	Configure Layer 3 VNI under VRF.

Configuring Anycast Gateway for VXLAN Routing

Procedure

	Command or Action	Purpose
Step 1	**fabric forwarding anycast-gateway-mac** *address*	Configure distributed gateway virtual MAC address. **Note** One virtual MAC per VTEP **Note** All VTEPs must have the same virtual MAC address.
Step 2	**fabric forwarding mode anycast-gateway**	Associate SVI with Anycast Gateway under VLAN configuration mode.

Configuring the NVE Interface and VNIs

Procedure

	Command or Action	Purpose
Step 1	**interface** *nve-interface*	Configure the NVE interface.
Step 2	**host-reachability protocol bgp**	This defines BGP as the mechanism for host reachability advertisement
Step 3	**member vni** *vni* **associate-vrf**	Add Layer-3 VNIs, one per tenant VRF, to the overlay. **Note** Required for VXLAN routing only.
Step 4	**member vni** *vni*	Add Layer 2 VNIs to the tunnel interface.
Step 5	**mcast-group** *address*	Configure the mcast group on a per-VNI basis

Configuring BGP on the VTEP

Procedure

	Command or Action	Purpose
Step 1	**router bgp** *number*	Configure BGP.
Step 2	**router-id** *address*	Specify router address.
Step 3	**neighbor** *address* **remote-as** *number*	Define MP-BGP neighbors. Under each neighbor define l2vpn evpn.

	Command or Action	Purpose
Step 4	**address-family ipv4 unicast**	Configure address family for IPv4.
Step 5	**address-family l2vpn evpn**	Configure address family Layer 2 VPN EVPN under the BGP neighbor. **Note** Address-family ipv4 evpn for vxlan host-based routing
Step 6	(Optional) **Allowas-in**	Allows duplicate AS numbers in the AS path. Configure this parameter on the leaf for eBGP when all leafs are using the same AS, but the spines have a different AS than leafs.
Step 7	**send-community extended**	Configures community for BGP neighbors.
Step 8	**vrf** *vrf-name*	Specify VRF.
Step 9	**address-family ipv4 unicast**	Configure address family for IPv4.
Step 10	**advertise** *l2vpn* **evpn**	Enable advertising EVPN routes.
Step 11	**address-family ipv6 unicast**	Configure address family for IPv6.
Step 12	**advertise** *l2vpn* **evpn**	Enable advertising EVPN routes. **Note** To disable advertisement for a VRF toward the EVPN, disable the VNI in NVE by entering the **no member vni** *vni* **associate-vrf** command in interface nve1. The *vni* is the VNI associated with that particular VRF.

Configuring RD and Route Targets for VXLAN Bridging

Procedure

	Command or Action	Purpose
Step 1	**evpn**	Configure VRF.
Step 2	**vni** *number* **l2**	**Note** Only Layer 2 VNIs need to be specified.
Step 3	**rd auto**	Define VRF RD (route distinguisher) to configure VRF context.
Step 4	**route-target import auto**	Define VRF Route Target and import policies.

	Command or Action	Purpose
Step 5	**route-target export auto**	Define VRF Route Target and export policies.

About RD Auto

The auto-derived Route Distinguisher (rd auto) is based on the Type 1 encoding format as described in IETF RFC 4364 section 4.2 https://tools.ietf.org/html/rfc4364#section-4.2. The Type 1 encoding allows a 4-byte administrative field and a 2-byte numbering field. Within Cisco NX-OS, the auto derived RD is constructed with the IP address of the BGP Router ID as the 4-byte administrative field (RID) and the internal VRF identifier for the 2-byte numbering field (VRF ID).

The 2-byte numbering field is always derived from the VRF, but results in a different numbering scheme depending on its use for the IP-VRF or the MAC-VRF:

- The 2-byte numbering field for the IP-VRF uses the internal VRF ID starting at 1 and increments. VRF IDs 1 and 2 are reserved for the default VRF and the management VRF respectively. The first custom defined IP VRF uses VRF ID 3.
- The 2-byte numbering field for the MAC-VRF uses the VLAN ID + 32767, which results in 32768 for VLAN ID 1 and incrementing.

Example auto-derived Route Distinguisher (RD)

- IP-VRF with BGP Router ID 192.0.2.1 and VRF ID 6 - RD 192.0.2.1:6
- MAC-VRF with BGP Router ID 192.0.2.1 and VLAN 20 - RD 192.0.2.1:32787

About Route-Target Auto

The auto-derived Route-Target (route-target import/export/both auto) is based on the Type 0 encoding format as described in IETF RFC 4364 section 4.2 (https://tools.ietf.org/html/rfc4364#section-4.2). IETF RFC 4364 section 4.2 describes the Route Distinguisher format and IETF RFC 4364 section 4.3.1 refers that it is desirable to use a similar format for the Route-Targets. The Type 0 encoding allows a 2-byte administrative field and a 4-byte numbering field. Within Cisco NX-OS, the auto derived Route-Target is constructed with the Autonomous System Number (ASN) as the 2-byte administrative field and the Service Identifier (VNI) for the 4-byte numbering field.

Examples of an auto derived Route-Target (RT):

- IP-VRF within ASN 65001 and L3VNI 50001 - Route-Target 65001:50001
- MAC-VRF within ASN 65001 and L2VNI 30001 - Route-Target 65001:30001

For Multi-AS environments, the Route-Targets must either be statically defined or rewritten to match the ASN portion of the Route-Targets.

Note Auto derived Route-Targets for a 4-byte ASN are not supported.

Configuring BGP for EVPN on the Spine

Procedure

	Command or Action	Purpose
Step 1	**route-map permitall permit 10**	Configure route-map. **Note** The route-map keeps the next-hop unchanged for EVPN routes. • Required for eBGP. • Optional for iBGP.
Step 2	**set ip next-hop unchanged**	Set next-hop address. **Note** The route-map keeps the next-hop unchanged for EVPN routes. • Required for eBGP. • Optional for iBGP. **Note** When two next hops are enabled, next hop ordering is not maintained. If one of the next hops is a VXLAN next hop and the other next hop is local reachable via FIB/AM/Hmm, the local next hop reachable via FIB/AM/Hmm is always taken irrespective of the order. Directly/locally connected next hops are always given priority over remotely connected next hops.
Step 3	**router bgp** *autonomous system number*	Specify BGP.
Step 4	**address-family l2vpn evpn**	Configure address family Layer 2 VPN EVPN under the BGP neighbor.
Step 5	**retain route-target all**	Configure retain route-target all under address-family Layer 2 VPN EVPN [global]. **Note** Required for eBGP. Allows the spine to retain and advertise all EVPN routes when there are no local VNI configured with matching import route targets.

	Command or Action	Purpose
Step 6	**neighbor** *address* **remote-as** *number*	Define neighbor.
Step 7	**address-family l2vpn evpn**	Configure address family Layer 2 VPN EVPN under the BGP neighbor.
Step 8	**disable-peer-as-check**	Disables checking the peer AS number during route advertisement. Configure this parameter on the spine for eBGP when all leafs are using the same AS but the spines have a different AS than leafs. **Note** Required for eBGP.
Step 9	**send-community extended**	Configures community for BGP neighbors.
Step 10	**route-map permitall out**	Applies route-map to keep the next-hop unchanged. **Note** Required for eBGP.

Suppressing ARP

Suppressing ARP includes changing the size of the ACL ternary content addressable memory (TCAM) regions in the hardware.

Procedure

	Command or Action	Purpose
Step 1	**hardware access-list tcam region arp-ether** *size* **double-wide**	Configure TCAM region to suppress ARP. *tcam-size*—TCAM size. The size has to be a multiple of 256. If the size is more than 256, it has to be a multiple of 512. **Note** Reload is required for the TCAM configuration to be in effect.
Step 2	**interface nve 1**	Create the network virtualization endpoint (NVE) interface.
Step 3	**member vni** *vni-id*	Specify VNI ID.
Step 4	**suppress-arp**	Configure to suppress ARP under Layer 2 VNI.
Step 5	**copy running-config start-up-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Disabling VXLANs

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters configuration mode.
Step 2	**no nv overlay evpn**	Disables EVPN control plane.
Step 3	**no feature vn-segment-vlan-based**	Disables the global mode for all VXLAN bridge domains
Step 4	**no feature nv overlay**	Disables the VXLAN feature.
Step 5	(Optional) **copy running-config startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Duplicate Detection for IP and MAC Addresses

Cisco NX-OS supports duplicate detection for IP and MAC addresses. This enables the detection of duplicate IP or MAC addresses based on the number of moves in a given time-interval (seconds).

The default is 5 moves in 180 seconds. (Default number of moves is 5 moves. Default time-interval is 180 seconds.)

- For IP addresses:
  - After the 5th move within 180 seconds, the switch starts a 30 second lock (hold down timer) before checking to see if the duplication still exists (an effort to prevent an increment of the sequence bit). This 30 second lock can occur 5 times within 24 hours (this means 5 moves in 180 seconds for 5 times) before the switch permanently locks or freezes the duplicate entry. (**show fabric forwarding ip local-host-db vrf abc**)
- For MAC addresses:
  - After the 5th move within 180 seconds, the switch starts a 30 second lock (hold down timer) before checking to see if the duplication still exists (an effort to prevent an increment of the sequence bit). This 30 second lock can occur 3 times within 24 hours (this means 5 moves in 180 seconds for 3 times) before the switch permanently locks or freezes the duplicate entry. (**show l2rib internal permanently-frozen-list**)
- Wherever a MAC address is permanently frozen, a syslog message with written by L2RIB.

```
2017 Jul  5 10:27:34 leaf %$ VDC-1 %$  %USER-2-SYSTEM_MSG: Unfreeze limit (3) hit, MAC
 0000.0033.3333in topo: 200 is permanently frozen - l2rib
2017 Jul  5 10:27:34 leaf %$ VDC-1 %$  %USER-2-SYSTEM_MSG: Detected duplicate host
0000.0033.3333, topology 200, during Local update, with host located at remote VTEP
1.2.3.4, VNI 2 - l2rib
2017 Jul  5 10:27:34 leaf %$ VDC-1 %$  %USER-2-SYSTEM_MSG: Unfreeze limit (3) hit, MAC
 0000.0033.3334in topo: 200 is permanently frozen - l2rib
2017 Jul  5 10:27:34 leaf %$ VDC-1 %$  %USER-2-SYSTEM_MSG: Detected duplicate host
```

```
0000.0033.3334, topology 200, during Local update, with host l
```

The following are example commands to help the configuration of the number of VM moves in a specific time interval (seconds) for duplicate IP-detection:

Command	Description

`switch(config)# fabric forwarding ?`
`anycast-gateway-mac`
`dup-host-ip-addr-detection`
	Available sub-commands:
	• Anycast gateway MAC of the switch.
	• To detect duplicate host addresses in n seconds.

`switch(config)# fabric forwarding`
`dup-host-ip-addr-detection ?`
`<1-1000>`
	The number of host moves allowed in n seconds. The range is 1 to 1000 moves; default is 5 moves.

`switch(config)# fabric forwarding`
`dup-host-ip-addr-detection 100 ?`
`<2-36000>`
	The duplicate detection timeout in seconds for the number of host moves. The range is 2 to 36000 seconds; default is 180 seconds.

`switch(config)# fabric forwarding` `dup-host-ip-addr-detection 100 10`	Detects duplicate host addresses (limited to 100 moves) in a period of 10 seconds.

The following are example commands to help the configuration of the number of VM moves in a specific time interval (seconds) for duplicate MAC-detection:

Command	Description

`switch(config)# l2rib dup-host-mac-detection ?`
`<1-1000>`
`default`
	Available sub-commands for L2RIB:
	• The number of host moves allowed in n seconds. The range is 1 to 1000 moves.
	• Default setting (5 moves in 180 in seconds).

`switch(config)# l2rib dup-host-mac-detection 100 ?` `<2-36000>`	The duplicate detection timeout in seconds for the number of host moves. The range is 2 to 36000 seconds; default is 180 seconds.
`switch(config)# l2rib dup-host-mac-detection 100 10`	Detects duplicate host addresses (limited to 100 moves) in a period of 10 seconds.

Enabling Nuage Controller Interoperability

The following steps enable Nuage controller interoperability.

Procedure

	Command or Action	Purpose
Step 1	**nuage controller interop**	Global command to enable interoperability mode.
Step 2	**router bgp** *number*	Configure BGP.
Step 3	**address-family l2vpn evpn**	Configure address family Layer 2 VPN EVPN under the BGP neighbor.
Step 4	**advertise-system-mac**	Enable Nuage interoperability mode for BGP.
Step 5	**allow-vni-in-ethertag**	Enable Nuage interoperability mode for BGP.
Step 6	**route-map permitall permit 10**	Configure route-map to permit all.
Step 7	**router bgp** *number*	Configure BGP.
Step 8	**vrf** *vrf-name*	Specify tenant VRF.
Step 9	**address-family ipv4 unicast**	Configure address family for IPv4.
Step 10	**advertise l2vpn evpn**	Enable advertising EVPN routes.
Step 11	**redistribute hmm route-map permitall**	Enables advertise host tenant routes as evpn type-5 routes for interoperability.

Example

The following is an example to enable Nuage controller interoperability:

```
/*** Enable interoperability mode at global level. ***/
switch(config)# nuage controller interop

/*** Configure BGP to enable interoperability mode. ***/
switch(config)# router bgp 1001
switch(config-router)# address-family l2vpn evpn
switch(config-router-af)# advertise-system-mac
switch(config-router-af)# allow-vni-in-ethertag

/*** Advertise host tenant routes as evpn type-5 routes for interoperability. ***/
switch(config)# route-map permitall permit 10
switch(config)# router bgp 1001
switch(config-router)# vrf vni-491830
switch(config-router-vrf)# address-family ipv4 unicast
switch(config-router-vrf-af)# advertise l2vpn evpn
switch(config-router-vrf-af)# redistribute hmm route-map permitall
```

Verifying the VXLAN BGP EVPN Configuration

To display the VXLAN BGP EVPN configuration information, enter one of the following commands:

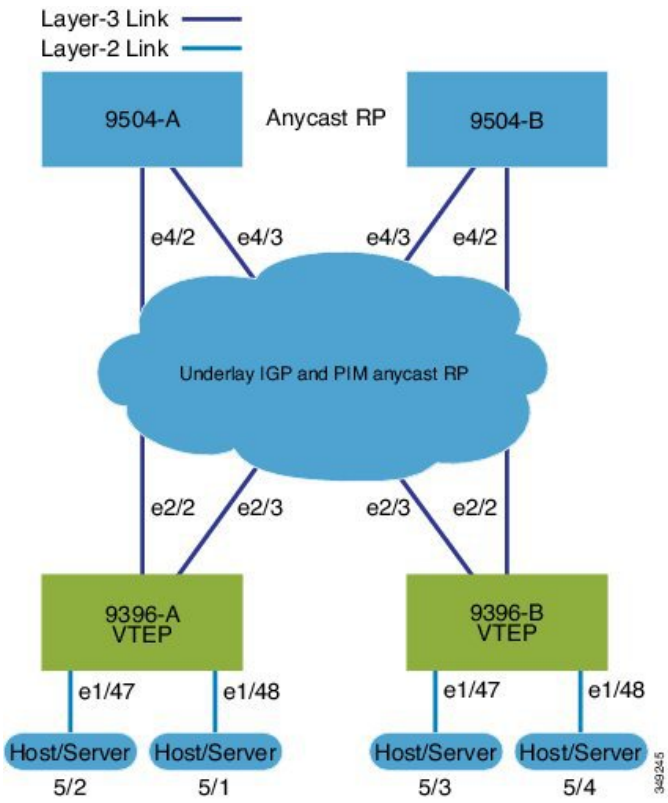
Command	Purpose
show nve vrf	Displays VRFs and associated VNIs
show bgp l2vpn evpn	Displays routing table information.
show ip arp suppression-cache [**detail** | **summary** | **vlan** *vlan* | **statistics**]	Displays ARP suppression information.
show vxlan interface	Displays VXLAN interface status.
show vxlan interface | count	Displays VXLAN VLAN logical port VP count. **Note** A VP is allocated on a per-port per-VLAN basis. The sum of all VPs across all VXLAN-enabled Layer 2 ports gives the total logical port VP count. For example, if there are 10 Layer 2 trunk interfaces, each with 10 VXLAN VLANs, then the total VXLAN VLAN logical port VP count is 10*10 = 100.
show l2route evpn mac [**all** | **evi** *evi* [**bgp** | **local** | **static** | **vxlan** | **arp**]]	Displays Layer 2 route information.
show l2route evpn fl all	Displays all fl routes.
show l2route evpn imet all	Displays all imet routes.
show l2route evpn mac-ip all **show l2route evpn mac-ip all detail**	Displays all MAC IP routes.
show l2route topology	Displays Layer 2 route topology.

Note Although the **show ip bgp** command is available for verifying a BGP configuration, as a best practice, it is preferable to use the **show bgp** command instead.

Example of VXLAN BGP EVPN (EBGP)

An example of a VXLAN BGP EVPN (EBGP):

Figure 9: VXLAN BGP EVPN Topology (EBGP)



EBGP between Spine and Leaf

- Spine (9504-A)
  - Enable the EVPN control plane

```
nv overlay evpn
```

  - Enable the relevant protocols

```
feature bgp
feature pim
```

  - Configure Loopback for BGP

```
interface loopback0
  ip address 10.1.1.1/32
  ip pim sparse-mode
```

  - Configure Loopback for Anycast RP

```
interface loopback1
  ip address 100.1.1.1/32
  ip pim sparse-mode
```

  - Configure Anycast RP

```
ip pim rp-address 100.1.1.1 group-list 224.0.0.0/4
```

```
ip pim ssm range 232.0.0.0/8
ip pim anycast-rp 100.1.1.1 10.1.1.1
ip pim anycast-rp 100.1.1.1 20.1.1.1
```

- Configure route-map used by EBGP for Spine

```
route-map permitall permit 10
  set ip next-hop unchanged
```

- Configure interfaces for Spine-leaf interconnect

```
interface Ethernet4/2
  ip address 192.168.1.42/24
  ip pim sparse-mode
  no shutdown

interface Ethernet4/3
  ip address 192.168.2.43/24
  ip pim sparse-mode
  no shutdown
```

- Configure the BGP overlay for the EVPN address family.

```
router bgp 100
  router-id 10.1.1.1
  address-family l2vpn evpn
    nexthop route-map permitall
    retain route-target all
  neighbor 30.1.1.1 remote-as 200
    update-source loopback0
    ebgp-multihop 3
    address-family l2vpn evpn
      disable-peer-as-check
      send-community extended
      route-map permitall out
  neighbor 40.1.1.1 remote-as 200
    update-source loopback0
    ebgp-multihop 3
    address-family l2vpn evpn
      disable-peer-as-check
      send-community extended
      route-map permitall out
```

- Configure the BGP underlay.

```
neighbor 192.168.1.43 remote-as 200
    address-family ipv4 unicast
      allowas-in
      disable-peer-as-check
```

- Spine (9504-B)
  - Enable the EVPN control plane and the relevant protocols

```
nv overlay evpn
feature bgp
feature pim
```

- Configure Anycast RP

```
ip pim rp-address 100.1.1.1 group-list 224.0.0.0/4
ip pim ssm range 232.0.0.0/8
ip pim anycast-rp 100.1.1.1 10.1.1.1
ip pim anycast-rp 100.1.1.1 20.1.1.1
```

- Configure route-map used by EBGP for Spine

```
route-map permitall permit 10
  set ip next-hop unchanged
```

- Configure interfaces for Spine-leaf interconnect

```
interface Ethernet4/2
  ip address 192.168.4.42/24
  ip pim sparse-mode
  no shutdown

interface Ethernet4/3
  ip address 192.168.3.43/24
  ip pim sparse-mode
  no shutdown
```

- Configure Loopback for BGP

```
interface loopback0
  ip address 20.1.1.1/32
  ip pim sparse-mode
```

- Configure Loopback for Anycast RP

```
interface loopback1
  ip address 100.1.1.1/32
  ip pim sparse-mode
```

- Configure the BGP overlay for the EVPN address family.

```
router bgp 100
  router-id 20.1.1.1
  address-family l2vpn evpn
    retain route-target all
  neighbor 30.1.1.1 remote-as 200
    update-source loopback0
    ebgp-multihop 3
    address-family l2vpn evpn
      disable-peer-as-check
      send-community extended
      route-map permitall out
  neighbor 40.1.1.1 remote-as 200
    ebgp-multihop 3
    address-family l2vpn evpn
      disable-peer-as-check
      send-community extended
      route-map permitall out
```

- Configure the BGP underlay.

```
neighbor 192.168.1.43 remote-as 200
    address-family ipv4 unicast
      allowas-in
      disable-peer-as-check
```

- Leaf (9396-A)
  - Enable the EVPN control plane

    ```
    nv overlay evpn
    ```

  - Enable the relevant protocols

    ```
    feature bgp
    feature pim
    feature interface-vlan
    ```

  - Enable VXLAN with distributed anycast-gateway using BGP EVPN

    ```
    feature vn-segment-vlan-based
    feature nv overlay
    fabric forwarding anycast-gateway-mac 0000.2222.3333
    ```

  - Enable PIM RP

    ```
    ip pim rp-address 100.1.1.1 group-list 224.0.0.0/4
    ip pim ssm range 232.0.0.0/8
    ```

  - Create VLANs

    ```
    vlan 1-1002
    ```

  - Configure Loopback for BGP

    ```
    interface loopback0
      ip address 30.1.1.1/32
      ip pim sparse-mode
    ```

  - Configure Loopback for local VTEP IP

    ```
    interface loopback1
      ip address 50.1.1.1/32
      ip pim sparse-mode
    ```

  - Configure interfaces for Spine-leaf interconnect

    ```
    interface Ethernet2/2
      ip address 192.168.1.22/24
      ip pim sparse-mode
      no shutdown

    interface Ethernet2/3
      ip address 192.168.3.23/24
      ip pim sparse-mode
    ```

```
  no shutdown
```

- Create the VRF overlay VLAN and configure the vn-segment.

```
vlan 101
  vn-segment 900001
```

- Configure VRF overlay VLAN/SVI for the VRF

```
interface Vlan101
  no shutdown
  vrf member vxlan-900001
  ip forward
```

- Create VLAN and provide mapping to VXLAN

```
vlan 1001
  vn-segment 2001001
vlan 1002
  vn-segment 2001002
```

- Create VRF and configure VNI

```
vrf context vxlan-900001
  vni 900001


rd auto
  address-family ipv4 unicast
    route-target import 65535:101 evpn
    route-target export 65535:101 evpn
    route-target import 65535:101
    route-target export 65535:101
  address-family ipv6 unicast
    route-target import 65535:101 evpn
    route-target export 65535:101 evpn
    route-target import 65535:101
    route-target export 65535:101
```

- Create server facing SVI and enable distributed anycast-gateway

```
interface Vlan1001
  no shutdown
  vrf member vxlan-900001
  ip address 4.1.1.1/24
  ipv6 address 4:1:0:1::1/64
  fabric forwarding mode anycast-gateway

interface Vlan1002
  no shutdown
  vrf member vxlan-900001
  ip address 4.2.2.1/24
  ipv6 address 4:2:0:1::1/64
  fabric forwarding mode anycast-gateway
```

- Configure ACL TCAM region for ARP suppression

```
hardware access-list tcam region arp-ether 256 double-wide
```

- Create the network virtualization endpoint (NVE) interface

```
interface nve1
  no shutdown
  source-interface loopback1
  host-reachability protocol bgp
  member vni 900001 associate-vrf
  member vni 2001001
  mcast-group 239.0.0.1
  member vni 2001002
  mcast-group 239.0.0.1
```

- Configure interfaces for hosts/servers.

```
interface Ethernet1/47
  switchport
  switchport access vlan 1002
interface Ethernet1/48
  switchport
  switchport access vlan 1001
```

- Configure BGP

```
router bgp 200
router-id 30.1.1.1
  neighbor 10.1.1.1 remote-as 100
    update-source loopback0
    ebgp-multihop 3
      allowas-in
      send-community extended
    address-family l2vpn evpn
      allowas-in
      send-community extended
  neighbor 20.1.1.1 remote-as 100
    update-source loopback0
    ebgp-multihop 3
      allowas-in
      send-community extended
    address-family l2vpn evpn
      allowas-in
      send-community extended
  vrf vxlan-900001



evpn
  vni 2001001 l2
  vni 2001002 l2

rd auto
route-target import auto
route-target export auto
```

- Leaf (9396-B)
  - Enable the EVPN control plane functionality and the relevant protocols

```
nv overlay evpn
feature bgp
feature pim
feature interface-vlan
feature vn-segment-vlan-based
feature nv overlay
```

- Enable PIM RP

```
ip pim rp-address 100.1.1.1 group-list 224.0.0.0/4
ip pim ssm range 232.0.0.0/8
```

- Enable VXLAN with distributed anycast-gateway using BGP EVPN

```
fabric forwarding anycast-gateway-mac 0000.2222.3333
```

- Create VLANs

```
vlan 1-1002
```

- Create the VRF overlay VLAN and configure the vn-segment

```
vlan 101
  vn-segment 900001
```

- Create VLAN and provide mapping to VXLAN

```
vlan 1001
  vn-segment 2001001
vlan 1002
  vn-segment 2001002
```

- Create VRF and configure VNI

```
vrf context vxlan-900001
  vni 900001



  rd auto
  address-family ipv4 unicast
    route-target import 65535:101 evpn
    route-target export 65535:101 evpn
    route-target import 65535:101
    route-target export 65535:101
  address-family ipv6 unicast
    route-target import 65535:101 evpn
    route-target export 65535:101 evpn
    route-target import 65535:101 evpn
    route-target export 65535:101 evpn
```

- Configure ACL TCAM region for ARP suppression

```
hardware access-list tcam region arp-ether 256 double-wide
```

- Configure internal control VLAN/SVI for the VRF

```
interface Vlan101
  no shutdown
  vrf member vxlan-900001
  ip forward
```

- Create server facing SVI and enable distributed anycast-gateway

```
interface Vlan1001
  no shutdown
  vrf member vxlan-900001
  ip address 4.1.1.1/24
  ipv6 address 4:1:0:1::1/64
  fabric forwarding mode anycast-gateway

interface Vlan1002
  no shutdown
  vrf member vxlan-900001
  ip address 4.2.2.1/24
  ipv6 address 4:2:0:1::1/64
  fabric forwarding mode anycast-gateway
```

- Create the network virtualization endpoint (NVE) interface

```
interface nve1
  no shutdown
  source-interface loopback1
  host-reachability protocol bgp
  member vni 900001 associate-vrf
  member vni 2001001
  mcast-group 239.0.0.1
  member vni 2001002
  mcast-group 239.0.0.1
```

- Configure interfaces for hosts/servers

```
interface Ethernet1/47
  switchport
  switchport access vlan 1002

interface Ethernet1/48
  switchport
  switchport access vlan 1001
```

- Configure interfaces for Spine-leaf interconnect

```
interface Ethernet2/2
  ip address 192.168.4.22/24
  ip pim sparse-mode
  no shutdown

interface Ethernet2/3
  ip address 192.168.2.23/24
  ip pim sparse-mode
  no shutdown
```

- Configure Loopback for BGP

```
interface loopback0
  ip address 40.1.1.1/32
  ip pim sparse-mode
```

- Configure Loopback for local VTEP IP

```
interface loopback1
  ip address 51.1.1.1/32
  ip pim sparse-mode
```

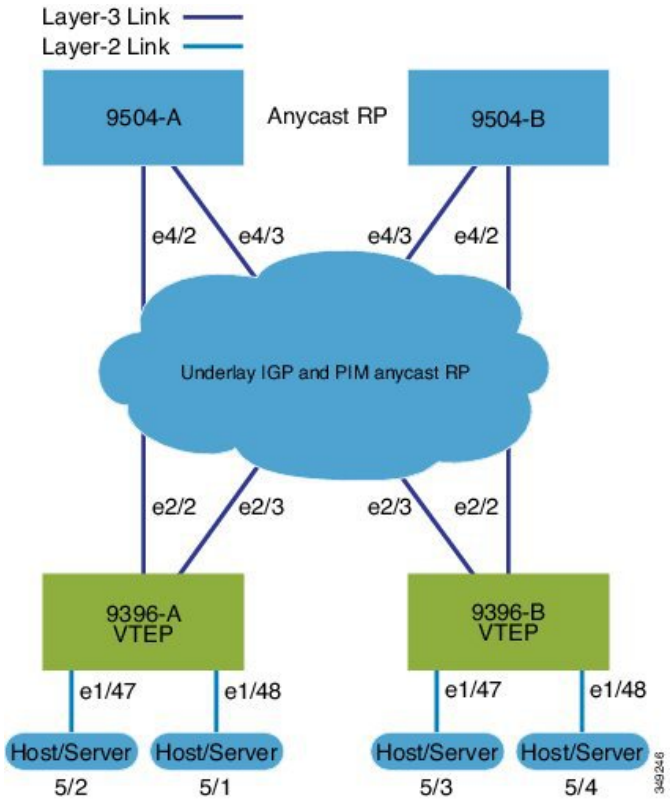
- Configure BGP

```
router bgp 200
router-id 40.1.1.1
  neighbor 10.1.1.1 remote-as 100
    update-source loopback0
    ebgp-multihop 3
      allowas-in
      send-community extended
    address-family l2vpn
      allowas-in
      send-community extended
  neighbor 20.1.1.1 remote-as 100
    update-source loopback0
    ebgp-multihop 3
      allowas-in
      send-community extended
    address-family l2vpn
      allowas-in
      send-community extended
  vrf vxlan-900001
```

Example of VXLAN BGP EVPN (IBGP)

An example of a VXLAN BGP EVPN (IBGP):

Figure 10: VXLAN BGP EVPN Topology (IBGP)



IBGP between Spine and Leaf

- Spine (9504-A)
  - Enable the EVPN control plane

```
nv overlay evpn
```

  - Enable the relevant protocols

```
feature ospf
feature bgp
feature pim
```

  - Configure Loopback for local VTEP IP, and BGP

```
interface loopback0
  ip address 10.1.1.1/32
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
```

  - Configure Loopback for Anycast RP

```
interface loopback1
  ip address 100.1.1.1/32
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
```

- Configure Anycast RP

```
ip pim rp-address 100.1.1.1 group-list 224.0.0.0/4
ip pim ssm range 232.0.0.0/8
ip pim anycast-rp 100.1.1.1 10.1.1.1
ip pim anycast-rp 100.1.1.1 20.1.1.1
```

- Enable OSPF for underlay routing

```
router ospf 1
```

- Configure interfaces for Spine-leaf interconnect

```
interface Ethernet4/2
  ip address 192.168.1.42/24
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
  no shutdown

interface Ethernet4/3
  ip address 192.168.2.43/24
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
  no shutdown
```

- Configure BGP

```
router bgp 65535
router-id 10.1.1.1
  neighbor 30.1.1.1 remote-as 65535
    update-source loopback0
    address-family l2vpn evpn
      send-community both
      route-reflector-client
  neighbor 40.1.1.1 remote-as 65535
    update-source loopback0
    address-family l2vpn evpn
      send-community both
      route-reflector-client
```

- Spine (9504-B)
  - Enable the EVPN control plane and the relevant protocols

```
nv overlay evpn
feature ospf
feature bgp
feature pim
```

  - Configure Anycast RP

```
ip pim rp-address 100.1.1.1 group-list 224.0.0.0/4
ip pim ssm range 232.0.0.0/8
ip pim anycast-rp 100.1.1.1 10.1.1.1
ip pim anycast-rp 100.1.1.1 20.1.1.1
```

  - Configure interfaces for Spine-leaf interconnect

```
interface Ethernet4/2
  ip address 192.168.4.42/24
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
  no shutdown

interface Ethernet4/3
  ip address 192.168.3.43/24
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
  no shutdown
```

- Configure Loopback for local VTEP IP, and BGP

```
interface loopback0
  ip address 20.1.1.1/32
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
```

- Configure Loopback for Anycast RP

```
interface loopback1
  ip address 100.1.1.1/32
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
```

- Enable OSPF for underlay routing

```
router ospf 1
```

- Configure BGP

```
router bgp 65535
router-id 20.1.1.1
  neighbor 30.1.1.1 remote-as 65535
    update-source loopback0
    address-family l2vpn evpn
      send-community both
      route-reflector-client
  neighbor 40.1.1.1 remote-as 65535
    update-source loopback0
     address-family l2vpn evpn
      send-community both
      route-reflector-client
```

- Leaf (9396-A)
  - Enable the EVPN control plane

```
nv overlay evpn
```

  - Enable the relevant protocols

```
feature ospf
feature bgp
feature pim
feature interface-vlan
```

- Enable VxLAN with distributed anycast-gateway using BGP EVPN

```
feature vn-segment-vlan-based
feature nv overlay
fabric forwarding anycast-gateway-mac 0000.2222.3333
```

- Enabling OSPF for underlay routing

```
router ospf 1
```

- Configure Loopback for local VTEP IP, and BGP

```
interface loopback0
  ip address 30.1.1.1/32
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
```

- Configure interfaces for Spine-leaf interconnect

```
interface Ethernet2/2
  no switchport
  ip address 192.168.1.22/24
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
  no shutdown

interface Ethernet2/3
  no switchport
  ip address 192.168.3.23/24
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
  no shutdown
```

- Configure PIM RP

```
ip pim rp-address 100.1.1.1 group-list 224.0.0.0/4
ip pim ssm range 232.0.0.0/8
```

- Create VLANs

```
vlan 1-1002
```

- Create overlay VRF VLAN and configure vn-segment

```
vlan 101
  vn-segment 900001
```

- Configure VRF overlay VLAN/SVI for the VRF

```
interface Vlan101
  no shutdown
  vrf member vxlan-900001
  ip forward
```

- Create VLAN and provide mapping to VXLAN

```
vlan 1001
  vn-segment 2001001
vlan 1002
  vn-segment 2001002
```

- Create VRF and configure VNI

```
vrf context vxlan-900001
  vni 900001


rd auto
  address-family ipv4 unicast
    route-target both auto
    route-target both auto evpn
  address-family ipv6 unicast
    route-target both auto
    route-target both auto evpn
```

- Create server facing SVI and enable distributed anycast-gateway

```
interface Vlan1001
  no shutdown
  vrf member vxlan-900001
  ip address 4.1.1.1/24
  ipv6 address 4:1:0:1::1/64
  fabric forwarding mode anycast-gateway

interface Vlan1002
  no shutdown
  vrf member vxlan-900001
  ip address 4.2.2.1/24
  ipv6 address 4:2:0:1::1/64
  fabric forwarding mode anycast-gateway
```

- Configure ACL TCAM region for ARP suppression

```
hardware access-list tcam region arp-ether 256 double-wide
```

- Create the network virtualization endpoint (NVE) interface

```
interface nve1
  no shutdown
  source-interface loopback0
  host-reachability protocol bgp
  member vni 900001 associate-vrf
  member vni 2001001
    mcast-group 239.0.0.1
  member vni 2001002
    mcast-group 239.0.0.1
```

- Configure interfaces for hosts/servers

```
interface Ethernet1/47
  switchport
  switchport access vlan 1002

interface Ethernet1/48
```

```
  switchport
  switchport access vlan 1001
```

- Configure BGP

```
router bgp 65535
router-id 30.1.1.1
  neighbor 10.1.1.1 remote-as 65535
    update-source loopback0
    address-family l2vpn evpn
      send-community both
  neighbor 20.1.1.1 remote-as 65535
    update-source loopback0
    address-family l2vpn evpn
      send-community both
  vrf vxlan-900001


evpn
  vni 2001001 l2
  vni 2001002 l2

rd auto
    route-target import auto
    route-target export auto
```

- Leaf (9396-B)
  - Enable the EVPN control plane functionality and the relevant protocols

```
nv overlay evpn
feature ospf
feature bgp
feature pim
feature interface-vlan
feature vn-segment-vlan-based
feature nv overlay
```

  - Enable VxLAN with distributed anycast-gateway using BGP EVPN

```
fabric forwarding anycast-gateway-mac 0000.2222.3333
```

  - Configure PIM RP

```
ip pim rp-address 100.1.1.1 group-list 224.0.0.0/4
ip pim ssm range 232.0.0.0/8
```

  - Create VLANs

```
vlan 1-1002
```

  - Create overlay VRF VLAN and configure vn-segment

```
vlan 101
  vn-segment 900001
```

  - Create VLAN and provide mapping to VXLAN

```
vlan 1001
  vn-segment 2001001
vlan 1002
  vn-segment 2001002
```

- Create VRF and configure VNI

```
vrf context vxlan-900001
  vni 900001


rd auto
  address-family ipv4 unicast
    route-target both auto
    route-target both auto evpn
  address-family ipv6 unicast
    route-target both auto
    route-target both auto evpn
```

- Configure ACL TCAM region for ARP suppression

```
hardware access-list tcam region arp-ether 256 double-wide
```

- Configure internal control VLAN/SVI for the VRF

```
interface Vlan101
  no shutdown
  vrf member vxlan-900001
  ip forward
```

- Create server facing SVI and enable distributed anycast-gateway

```
interface Vlan1001
  no shutdown
  vrf member vxlan-900001
  ip address 4.1.1.1/24
  ipv6 address 4:1:0:1::1/64
  fabric forwarding mode anycast-gateway

interface Vlan1002
  no shutdown
  vrf member vxlan-900001
  ip address 4.2.2.1/24
  ipv6 address 4:2:0:1::1/64
  fabric forwarding mode anycast-gateway
```

- Create the network virtualization endpoint (NVE) interface

```
interface nve1
  no shutdown
  source-interface loopback0
  host-reachability protocol bgp
  member vni 900001 associate-vrf
  member vni 2001001
    mcast-group 239.0.0.1
  member vni 2001002
    mcast-group 239.0.0.1
```

- Configure interfaces for hosts/servers

```
interface Ethernet1/47
  switchport
  switchport access vlan 1002

interface Ethernet1/48
  switchport
  switchport access vlan 1001
```

- Configure interfaces for Spine-leaf interconnect

```
interface Ethernet2/2
  no switchport
  ip address 192.168.4.22/24
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
  no shutdown

interface Ethernet2/3
  no switchport
  ip address 192.168.2.23/24
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
  no shutdown
```

- Configure Loopback for local VTEP IP, and BGP

```
interface loopback0
  ip address 40.1.1.1/32
  ip router ospf 1 area 0.0.0.0
  ip pim sparse-mode
```

- Enabling OSPF for underlay routing

```
router ospf 1
```

- Configure BGP

```
router bgp 65535
router-id 40.1.1.1
  neighbor 10.1.1.1 remote-as 65535
    update-source loopback0
    address-family l2vpn evpn
      send-community both
  neighbor 20.1.1.1 remote-as 65535
    update-source loopback0
    address-family l2vpn evpn
      send-community both
  vrf vxlan-900001


evpn
  vni 2001001 l2
    rd auto
    route-target import auto
    route-target export auto
  vni 2001002 l2
    rd auto
```

```
route-target import auto
route-target export auto
```

Example Show Commands

- **show nve peers**

```
9396-B# show nve peers
Interface Peer-IP          Peer-State
--------- ---------------  ----------
nve1      30.1.1.1         Up
```

- **show nve vni**

```
9396-B# show nve vni
Codes: CP - Control Plane        DP - Data Plane
       UC - Unconfigured         SA - Suppress ARP

Interface VNI      Multicast-group   State Mode Type [BD/VRF]      Flags
--------- -------- ----------------- ----- ---- ------------------ -----
nve1      900001   n/a               Up    CP   L3 [vxlan-900001]
nve1      2001001  225.4.0.1         Up    CP   L2 [1001]          SA
nve1      2001002  225.4.0.1         Up    CP   L2 [1002]          SA
```

- **show ip arp suppression-cache detail**

```
9396-B# show ip arp suppression-cache detail

Flags: + - Adjacencies synced via CFSoE
       L - Local Adjacency
       R - Remote Adjacency
       L2 - Learnt over L2 interface

Ip Address      Age      Mac Address    Vlan Physical-ifindex    Flags

4.1.1.54        00:06:41 0054.0000.0000 1001 Ethernet1/48        L
4.1.1.51        00:20:33 0051.0000.0000 1001 (null)              R
4.2.2.53        00:06:41 0053.0000.0000 1002 Ethernet1/47        L
4.2.2.52        00:20:33 0052.0000.0000 1002 (null)              R
```

- **show vxlan interface**

```
9396-B# show vxlan interface
Interface       Vlan    VPL Ifindex     LTL             HW VP
=========       ====    ===========     ===             =====
Eth1/47         1002    0x4c07d22e      0x10000         5697
Eth1/48         1001    0x4c07d02f      0x10001         5698
```

- **show bgp l2vpn evpn summary**

```
9396-B# show bgp l2vpn evpn summary
BGP summary information for VRF default, address family L2VPN EVPN
```

```
BGP router identifier 40.1.1.1, local AS number 65535
BGP table version is 27, L2VPN EVPN config peers 2, capable peers 2
14 network entries and 18 paths using 2984 bytes of memory
BGP attribute entries [14/2240], BGP AS path entries [0/0]
BGP community entries [0/0], BGP clusterlist entries [2/8]

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
10.1.1.1        4 65535   30199   30194       27    0    0     2w6d 4
20.1.1.1        4 65535   30199   30194       27    0    0     2w6d 4
```

- **show bgp l2vpn evpn**

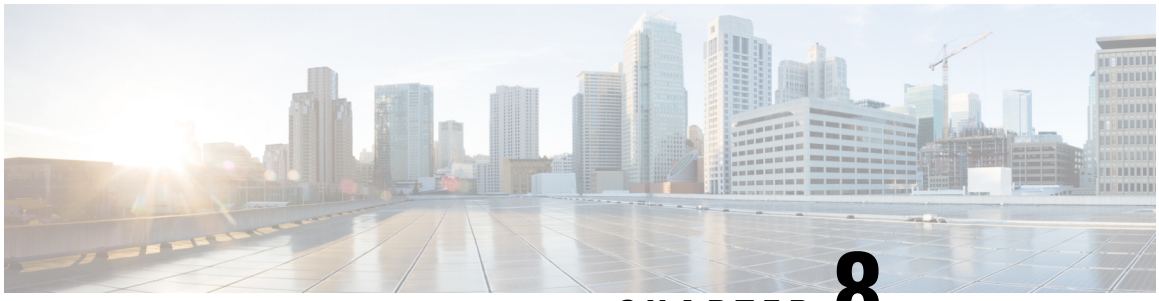
```
9396-B# show bgp l2vpn evpn
BGP routing table information for VRF default, address family L2VPN EVPN
BGP table version is 27, Local Router ID is 40.1.1.1
Status: s-suppressed, x-deleted, S-stale, d-dampened, h-history, *-valid, >-best
Path type: i-internal, e-external, c-confed, l-local, a-aggregate, r-redist, I-i
njected
Origin codes: i - IGP, e - EGP, ? - incomplete, | - multipath, & - backup

   Network            Next Hop            Metric     LocPrf     Weight Path
Route Distinguisher: 30.1.1.1:33768
*>i[2]:[0]:[0]:[48]:[d8b1.9071.e903]:[0]:[0.0.0.0]/216
                      30.1.1.1                           100          0 i
* i                   30.1.1.1                           100          0 i
*>i[2]:[0]:[0]:[48]:[d8b1.9071.e903]:[32]:[4.1.1.12]/272
                      30.1.1.1                           100          0 i
* i                   30.1.1.1                           100          0 i

Route Distinguisher: 30.1.1.1:33769
*>i[2]:[0]:[0]:[48]:[d8b1.9071.e903]:[0]:[0.0.0.0]/216
                      30.1.1.1                           100          0 i
* i                   30.1.1.1                           100          0 i
*>i[2]:[0]:[0]:[48]:[d8b1.9071.e903]:[32]:[4.2.2.11]/272
                      30.1.1.1                           100          0 i
* i                   30.1.1.1                           100          0 i

Route Distinguisher: 40.1.1.1:33768    (L2VNI 2001001)
*>i[2]:[0]:[0]:[48]:[d8b1.9071.e903]:[0]:[0.0.0.0]/216
                      30.1.1.1                           100          0 i
*>l[2]:[0]:[0]:[48]:[f8c2.8890.2a45]:[0]:[0.0.0.0]/216
                      40.1.1.1                           100      32768 i
*>i[2]:[0]:[0]:[48]:[d8b1.9071.e903]:[32]:[4.1.1.12]/272
                      30.1.1.1                           100          0 i
*>l[2]:[0]:[0]:[48]:[f8c2.8890.2a45]:[32]:[4.1.1.122]/272
                      40.1.1.1                           100      32768 i

Route Distinguisher: 40.1.1.1:33769    (L2VNI 2001002)
*>i[2]:[0]:[0]:[48]:[d8b1.9071.e903]:[0]:[0.0.0.0]/216
                      30.1.1.1                           100          0 i
*>l[2]:[0]:[0]:[48]:[f8c2.8890.2a45]:[0]:[0.0.0.0]/216
                      40.1.1.1                           100      32768 i
*>i[2]:[0]:[0]:[48]:[d8b1.9071.e903]:[32]:[4.2.2.11]/272
                      30.1.1.1                           100          0 i
*>l[2]:[0]:[0]:[48]:[f8c2.8890.2a45]:[32]:[4.2.2.111]/272
                      40.1.1.1                           100      32768 i

Route Distinguisher: 40.1.1.1:3    (L3VNI 900001)
*>i[2]:[0]:[0]:[48]:[d8b1.9071.e903]:[32]:[4.1.1.12]/272
                      30.1.1.1                           100          0 i
*>i[2]:[0]:[0]:[48]:[d8b1.9071.e903]:[32]:[4.2.2.11]/272
                      30.1.1.1                           100          0 i
```

- **show l2route evpn mac all**

```
9396-B# show l2route evpn mac all
Flags -(Rmac):Router MAC (Stt):Static (L):Local (R):Remote (V):vPC link
(Dup):Duplicate (Spl):Split (Rcv):Recv (AD):Auto-Delete (D):Del Pending
(S):Stale (C):Clear, (Ps):Peer Sync (O):Re-Originated (Nho):NH-Override
(Pf):Permanently-Frozen

Topology    Mac Address    Prod   Flags         Seq No     Next-Hops
----------- -------------- ------ ------------- ---------- ----------------
101         6412.2574.9f27 VXLAN  Rmac          0          30.1.1.1
1001        d8b1.9071.e903 BGP    SplRcv        0          30.1.1.1
1001        f8c2.8890.2a45 Local  L,            0          Eth1/48
1002        d8b1.9071.e903 BGP    SplRcv        0          30.1.1.1
1002        f8c2.8890.2a45 Local  L,            0          Eth1/47
```

- **show l2route evpn mac-ip all**

```
9396-B# show l2route evpn mac-ip all
Flags -(Rmac):Router MAC (Stt):Static (L):Local (R):Remote (V):vPC link
(Dup):Duplicate (Spl):Split (Rcv):Recv(D):Del Pending (S):Stale (C):Clear
(Ps):Peer Sync (Ro):Re-Originated
Topology    Mac Address    Prod   Flags         Seq No     Host IP         Next-Hops
----------- -------------- ------ ---------- --------------- ---------------
1001        d8b1.9071.e903 BGP    --            0          4.1.1.12       30.1.1.1
1001        f8c2.8890.2a45 HMM    --            0          4.1.1.122      Local
1002        d8b1.9071.e903 BGP    --            0          4.2.2.11       30.1.1.1
1002        f8c2.8890.2a45 HMM    --            0          4.2.2.111      Local
```



CHAPTER 8

Configuring VXLAN OAM

This chapter contains the following sections:

- VXLAN OAM Overview, on page 179
- Loopback (Ping) Message, on page 180
- Traceroute or Pathtrace Message, on page 181
- Configuring VXLAN OAM, on page 183
- Configuring NGOAM Profile, on page 186
- NGOAM Authentication, on page 187

VXLAN OAM Overview

The VXLAN operations, administration, and maintenance (OAM) protocol is a protocol for installing, monitoring, and troubleshooting Ethernet networks to enhance management in VXLAN based overlay networks.

Similar to ping, traceroute, or pathtrace utilities that allow quick determination of the problems in the IP networks, equivalent troubleshooting tools have been introduced to diagnose the problems in the VXLAN networks. The VXLAN OAM tools, for example, ping, pathtrace, and traceroute provide the reachability information to the hosts and the VTEPs in a VXLAN network. The OAM channel is used to identify the type of the VXLAN payload that is present in these OAM packets.

There are two types of payloads supported:

- Conventional ICMP packet to the destination to be tracked
- Special NVO3 draft Tissa OAM header that carries useful information

The ICMP channel helps to reach the traditional hosts or switches that do not support the new OAM packet formats. The NVO3 draft Tissa channels helps to reach the supported hosts or switches and carries the important diagnostic information. The VXLAN NVO3 draft Tissa OAM messages may be identified via the reserved OAM EtherType or by using a well-known reserved source MAC address in the OAM packets depending on the implementation on different platforms. This constitutes a signature for recognition of the VXLAN OAM packets. The VXLAN OAM tools are categorized as shown in table below.

Table 9: VXLAN OAM Tools

Category	Tools
Fault Verification	Loopback Message

Category	Tools
Fault Isolation	Path Trace Message
Performance	Delay Measurement, Loss Measurement
Auxiliary	Address Binding Verification, IP End Station Locator, Error Notification, OAM Command Messages, and Diagnostic Payload Discovery for ECMP Coverage

Loopback (Ping) Message

The loopback message (The ping and the loopback messages are the same and they are used interchangeably in this guide) is used for the fault verification. The loopback message utility is used to detect various errors and the path failures. Consider the topology in the following example where there are three core (spine) switches labeled Spine 1, Spine 2, and Spine 3 and five leaf switches connected in a Clos topology. The path of an example loopback message initiated from Leaf 1 for Leaf 5 is displayed when it traverses via Spine 3. When the loopback message initiated by Leaf 1 reaches Spine 3, it forwards it as VXLAN encapsulated data packet based on the outer header. The packet is not sent to the software on Spine 3. On Leaf 3, based on the appropriate loopback message signature, the packet is sent to the software VXLAN OAM module, that in turn, generates a loopback response that is sent back to the originator Leaf 1.

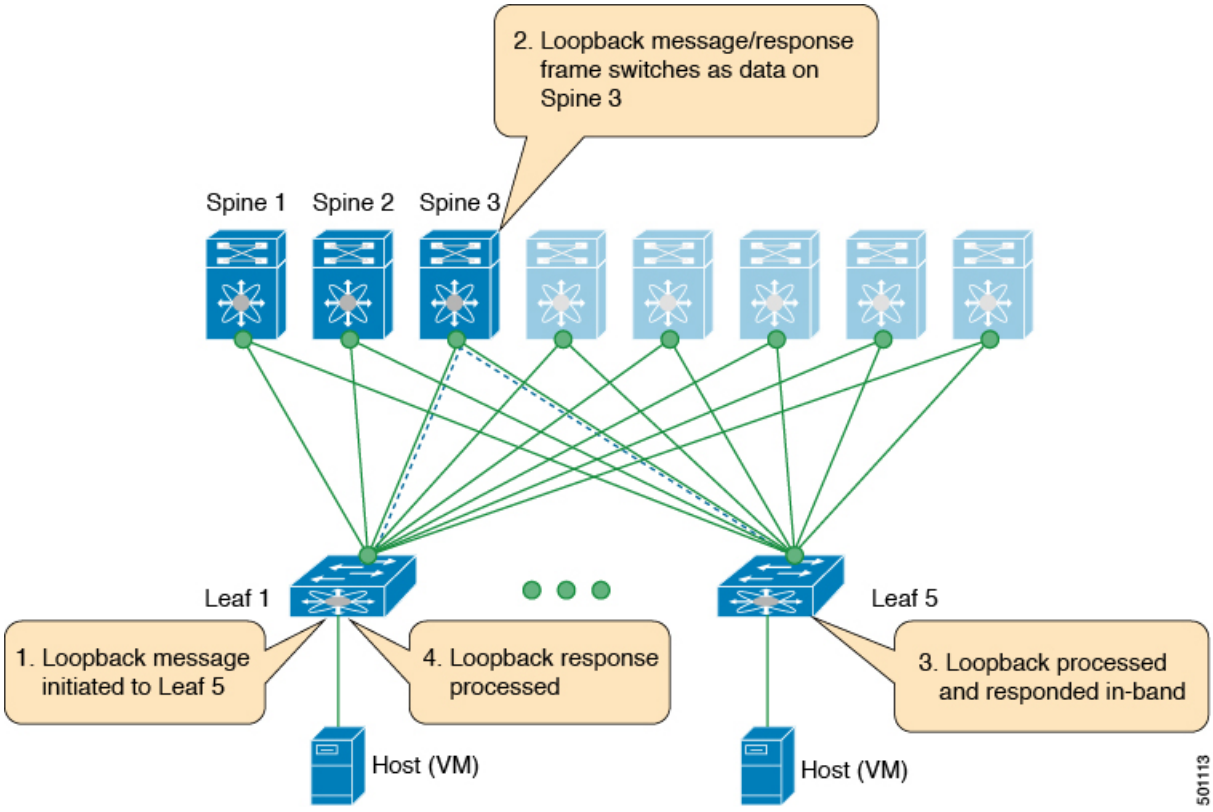
The loopback (ping) message can be destined to VM or to the (VTEP on) leaf switch. This ping message can use different OAM channels. If the ICMP channel is used, the loopback message can reach all the way to the VM if the VM's IP address is specified. If NVO3 draft Tissa channel is used, this loopback message is terminated on the leaf switch that is attached to the VM, as the VMs do not support the NVO3 draft Tissa headers in general. In that case, the leaf switch replies back to this message indicating the reachability of the VM. The ping message supports the following reachability options:

Ping

Check the network reachability (**Ping** command):

- From Leaf 1 (VTEP 1) to Leaf 2 (VTEP 2) (ICMP or NVO3 draft Tissa channel)
- From Leaf 1 (VTEP 1) to VM 2 (host attached to another VTEP) (ICMP or NVO3 draft Tissa channel)

Figure 11: Loopback Message



Traceroute or Pathtrace Message

The traceroute or pathtrace message is used for the fault isolation. In a VXLAN network, it may be desirable to find the list of switches that are traversed by a frame to reach the destination. When the loopback test from a source switch to a destination switch fails, the next step is to find out the offending switch in the path. The operation of the path trace message begins with the source switch transmitting a VXLAN OAM frame with a TTL value of 1. The next hop switch receives this frame, decrements the TTL, and on finding that the TTL is 0, it transmits a TTL expiry message to the sender switch. The sender switch records this message as an indication of success from the first hop switch. Then the source switch increases the TTL value by one in the next path trace message to find the second hop. At each new transmission, the sequence number in the message is incremented. Each intermediate switch along the path decrements the TTL value by 1 as is the case with regular VXLAN forwarding.

This process continues until a response is received from the destination switch, or the path trace process timeout occurs, or the hop count reaches a maximum configured value. The payload in the VXLAN OAM frames is referred to as the flow entropy. The flow entropy can be populated so as to choose a particular path among multiple ECMP paths between a source and destination switch. The TTL expiry message may also be generated by the intermediate switches for the actual data frames. The same payload of the original path trace request is preserved for the payload of the response.

The traceroute and pathtrace messages are similar, except that traceroute uses the ICMP channel, whereas pathtrace use the NVO3 draft Tissa channel. Pathtrace uses the NVO3 draft Tissa channel, carrying additional diagnostic information, for example, interface load and statistics of the hops taken by these messages. If an

intermediate device does not support the NVO3 draft Tissa channel, the pathtrace behaves as a simple traceroute and it provides only the hop information.

Traceroute

Trace the path that is traversed by the packet in the VXLAN overlay using **Traceroute** command:

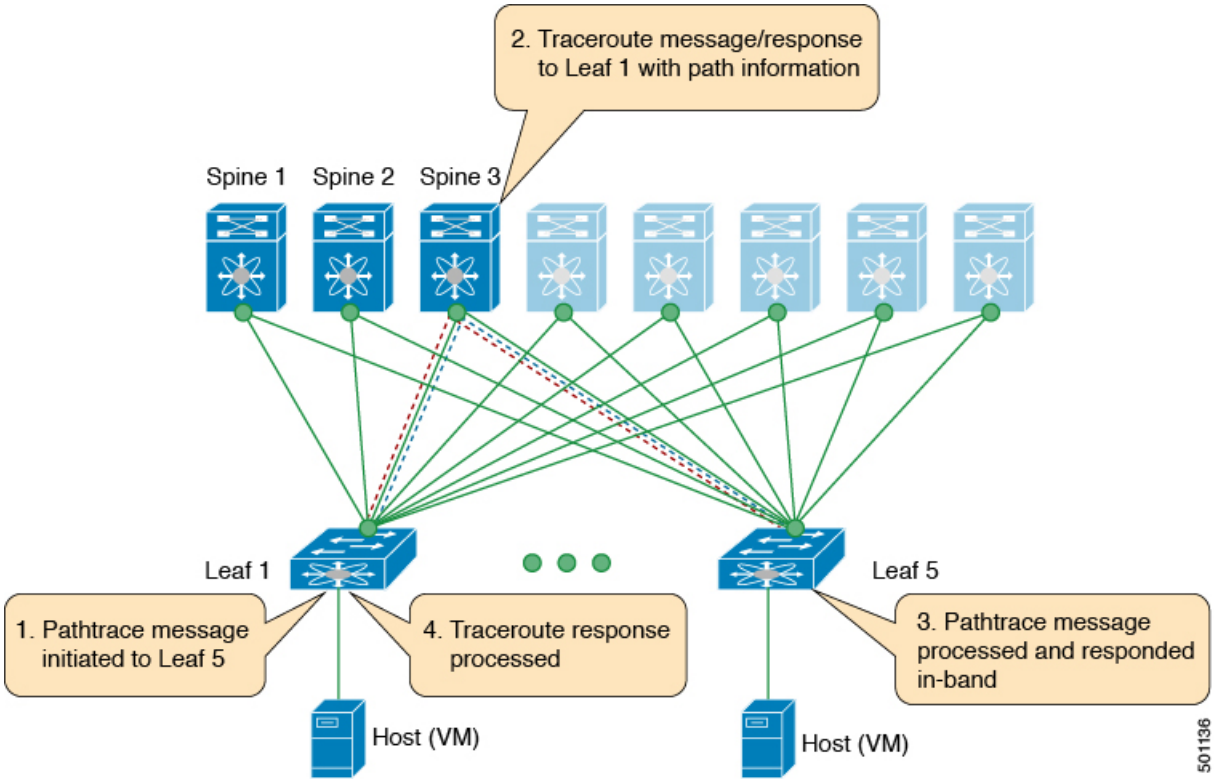
- Traceroute uses the ICMP packets (channel-1), encapsulated in the VXLAN encapsulation to reach the host

Pathtrace

Trace the path that is traversed by the packet in the VXLAN overlay using the NVO3 draft Tissa channel with **Pathtrace** command:

- Pathtrace uses special control packets like NVO3 draft Tissa or TISSA (channel-2) to provide additional information regarding the path (for example, ingress interface and egress interface). These packets terminate at VTEP and they does not reach the host. Therefore, only the VTEP responds.

Figure 12: Traceroute Message



Configuring VXLAN OAM

Before you begin

As a prerequisite, ensure that the VXLAN configuration is complete.

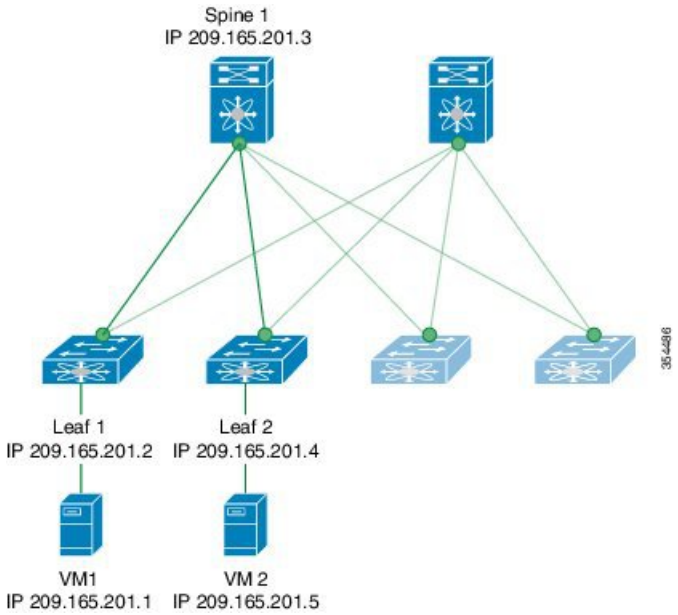
Procedure

	Command or Action	Purpose
Step 1	switch(config)# **feature ngoam**	Enters the NGOAM feature.
Step 2	switch(config)# **hardware access-list tcam region arp-ether 256 double-wide**	For Cisco Nexus 3000 Series switches with Network Forwarding Engine (NFE), configure the TCAM region for ARP-ETHER using this command. This step is essential to program the ACL rule in the hardware and it is a pre-requisite before installing the ACL rule. **Note** Configuring the TCAM region requires the node to be rebooted.
Step 3	switch(config)# **ngoam install acl**	Installs NGOAM Access Control List (ACL).
Step 4	(Optional) **#bcm-shell module 1 "fp show group 62"**	For Cisco Nexus 3000 Series switches with Network Forwarding Engine (NFE), complete this verification step. After entering the command, perform a lookup for entry/eid with data=0x8902 under EtherType.

Example

See the following examples of the configuration topology.

Figure 13: VXLAN Network



VXLAN OAM provides the visibility of the host at the switch level, that allows a leaf to ping the host using the **ping nve** command.

The following example displays how to ping from Leaf 1 to VM2 via Spine 1.

```
switch# ping nve ip 209.165.201.5 vrf vni-31000 source 1.1.1.1 verbose

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
'D' - Destination Unreachable, 'X' - unknown return code,
'm' - malformed request(parameter problem),
'c' - Corrupted Data/Test, '#' - Duplicate response


Sender handle: 34
! sport 40673 size 39,Reply from 209.165.201.5,time = 3 ms
! sport 40673 size 39,Reply from 209.165.201.5,time = 1 ms
! sport 40673 size 39,Reply from 209.165.201.5,time = 1 ms
! sport 40673 size 39,Reply from 209.165.201.5,time = 1 ms
! sport 40673 size 39,Reply from 209.165.201.5,time = 1 ms
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/4/18 ms
Total time elapsed 49 ms
```

Note The source ip-address 1.1.1.1 used in the above example is a loopback interface that is configured on Leaf 1 in the same VRF as the destination ip-address. For example, the VRF in this example is vni-31000.

The following example displays how to traceroute from Leaf 1 to VM 2 via Spine 1.

```
switch# traceroute nve ip 209.165.201.5 vrf vni-31000 source 1.1.1.1 verbose
```

```
Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
'D' - Destination Unreachable, 'X' - unknown return code,
'm' - malformed request(parameter problem),
'c' - Corrupted Data/Test, '#' - Duplicate response


Traceroute request to peer ip 209.165.201.4 source ip 209.165.201.2
Sender handle: 36
  1 !Reply from 209.165.201.3,time = 1 ms
  2 !Reply from 209.165.201.4,time = 2 ms
  3 !Reply from 209.165.201.5,time = 1 ms
```

The following example displays how to pathtrace from Leaf 2 to Leaf 1.

```
switch# pathtrace nve ip 209.165.201.4 vni 31000 verbose

Path trace Request to peer ip 209.165.201.4 source ip 209.165.201.2

Sender handle: 42
TTL   Code  Reply                IngressI/f    EgressI/f    State
=============================================================================
1     !Reply from 209.165.201.3,  Eth5/5/1      Eth5/5/2     UP/UP
2     !Reply from 209.165.201.4,  Eth1/3        Unknown      UP/DOWN
```

The following example displays how to MAC ping from Leaf 2 to Leaf 1 using NVO3 draft Tissa channel:

```
switch# ping nve mac 0050.569a.7418 2901 ethernet 1/51 profile 4 verbose

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
'D' - Destination Unreachable, 'X' - unknown return code,
'm' - malformed request(parameter problem),
'c' - Corrupted Data/Test, '#' - Duplicate response

Sender handle: 408
!!!!Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/5 ms
Total time elapsed 104 ms

switch# show run ngoam
feature ngoam
ngoam profile 4
oam-channel 2
ngoam install acl
```

The following example displays how to pathtrace based on a payload from Leaf 2 to Leaf 1:

```
switch# pathtrace nve ip unknown vrf vni-31000 payload mac-addr 0050.569a.d927 0050.569a.a4fa
ip 209.165.201.5 209.165.201.1 port 15334 12769 proto 17 payload-end

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
'D' - Destination Unreachable, 'X' - unknown return code,
'm' - malformed request(parameter problem),
'c' - Corrupted Data/Test, '#' - Duplicate response

Path trace Request to peer ip 209.165.201.4 source ip 209.165.201.2
Sender handle: 46
TTL Code Reply IngressI/f EgressI/f State
=============================================================================
```

```
1 !Reply from 209.165.201.3, Eth5/5/1 Eth5/5/2 UP/UP
2 !Reply from 209.165.201.4, Eth1/3 Unknown UP/DOWN
```

Configuring NGOAM Profile

Complete the following steps to configure NGOAM profile.

Procedure

	Command or Action	Purpose
Step 1	switch(config)#**[no] feature ngoam**	Enables or disables NGOAM feature
Step 2	switch(config)#**[no] ngoam profile <profile-id>**	Configures OAM profile. The range for the profile-id is <1 – 1023>. This command does not have a default value. Enters the config-ngoam-profile submode to configure NGOAM specific commands. **Note** All profiles have default values and the **show run all** CLI command displays them. The default values are not visible through the **show run** CLI command.

Step 3	switch(config-ng-oam-profile)# **?**
	Example:
	`switch(config-ng-oam-profile)# ?`
	`  description  Configure description of the profile`
	`  dot1q  Encapsulation dot1q/bd`
	`  flow  Configure ngoam flow`
	`  hop  Configure ngoam hop count`
	`  interface  Configure ngoam egress interface`
	`  no  Negate a command or set its defaults`
	`  oam-channel  Oam-channel used`
	`  payload  Configure ngoam payload`
	`  sport  Configure ngoam Udp source port range`
		Displays the options for configuring NGOAM profile.

Example

See the following examples for configuring an NGOAM profile and for configuring NGOAM flow.

```
switch(config)#
ngoam profile 1
oam-channel 1
flow forward
payload pad 0x2
sport 12345, 54321

switch(config-ngoam-profile)#flow {forward }
Enters config-ngoam-profile-flow submode to configure forward flow entropy specific
information
```

NGOAM Authentication

NGOAM provides the interface statistics in the pathtrace response. Beginning with Cisco NX-OS Release 7.0(3)I6(1), NGOAM authenticates the pathtrace requests to provide the statistics by using the HMAC MD5 authentication mechanism.

NGOAM authentication validates the pathtrace requests before providing the interface statistics. NGOAM authentication takes effect only for the pathtrace requests with **req-stats** option. All the other commands are not affected with the authentication configuration. If NGOAM authentication key is configured on the requesting node, NGOAM runs the MD5 algorithm using this key to generate the 16-bit MD5 digest. This digest is encoded as type-length-value (TLV) in the pathtrace request messages.

When the pathtrace request is received, NGOAM checks for the **req-stats** option and the local NGOAM authentication key. If the local NGOAM authentication key is present, it runs MD5 using the local key on the request to generate the MD5 digest. If both digests match, it includes the interface statistics. If both digests do not match, it sends only the interface names. If an NGOAM request comes with the MD5 digest but no local authentication key is configured, it ignores the digest and sends all the interface statistics. To secure an entire network, configure the authentication key on all nodes.

To configure the NGOAM authentication key, use the **ngoam authentication-key** <*key*> CLI command. Use the **show running-config ngoam** CLI command to display the authentication key.

```
switch# show running-config ngoam
!Time: Tue Mar 28 18:21:50 2017
version 7.0(3)I6(1)
feature ngoam
ngoam profile 1
  oam-channel 2
ngoam profile 3
ngoam install acl
ngoam authentication-key 987601ABCDEF
```

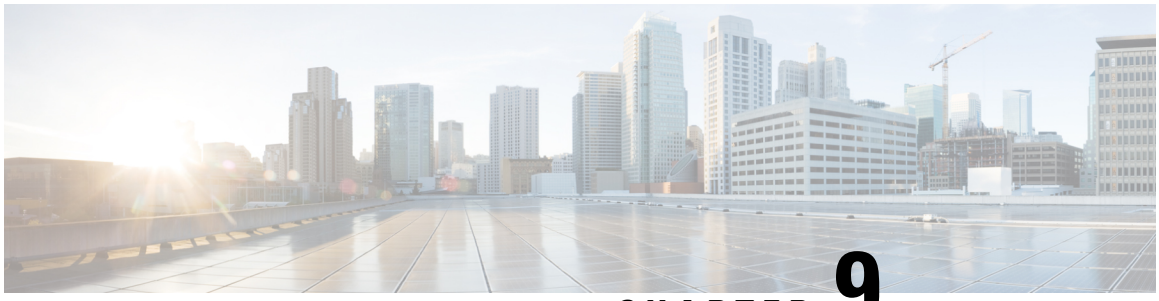
In the following example, the same authentication key is configured on the requesting switch and the responding switch.

```
switch# pathtrace nve ip 12.0.22.1 profile 1 vni 31000 req-stats ver
Path trace Request to peer ip 12.0.22.1 source ip 11.0.22.1
Hop   Code   ReplyIP   IngressI/f  EgressI/f   State
================================================
  1 !Reply from 55.55.55.2, Eth5/7/1  Eth5/7/2  UP / UP
  Input Stats: PktRate:0 ByteRate:0 Load:0 Bytes:339573434 unicast:14657 mcast:307581
bcast:67 discards:0 errors:3 unknown:0 bandwidth:42949672970000000
Output Stats: PktRate:0 ByteRate:0 load:0 bytes:237399176 unicast:2929 mcast:535710
```

```
bcast:10408 discards:0 errors:0 bandwidth:42949672970000000
  2 !Reply from 12.0.22.1, Eth1/7  Unknown  UP / DOWN
  Input Stats: PktRate:0 ByteRate:0 Load:0 Bytes:4213416 unicast:275 mcast:4366 bcast:3
discards:0 errors:0 unknown:0 bandwidth:42949672970000000
switch# conf t
switch(config)# no ngoam authentication-key 123456789
switch(config)# end
```

In the following example, an authentication key is not configured on the requesting switch. Therefore, the responding switch does not send any interface statistics. The intermediate node does not have any authentication key configured and it always replies with the interface statistics.

```
switch# pathtrace nve ip 12.0.22.1 profile 1 vni 31000 req-stats ver
Path trace Request to peer ip 12.0.22.1 source ip 11.0.22.1
Sender handle: 10
Hop   Code   ReplyIP   IngressI/f  EgressI/f   State
=================================================
  1 !Reply from 55.55.55.2, Eth5/7/1  Eth5/7/2  UP / UP
  Input Stats: PktRate:0 ByteRate:0 Load:0 Bytes:339580108 unicast:14658 mcast:307587
bcast:67 discards:0 errors:3 unknown:0 bandwidth:42949672970000000
Output Stats: PktRate:0 ByteRate:0 load:0 bytes:237405790 unicast:2929 mcast:535716
bcast:10408 discards:0 errors:0 bandwidth:42949672970000000
  2 !Reply from 12.0.22.1, Eth1/17  Unknown  UP / DOWN
```



CHAPTER 9

Configuring VXLAN Multihoming

•
• VXLAN EVPN Multihoming Overview , on page 189
• Configuring VXLAN EVPN Multihoming, on page 192
• Configuring Layer 2 Gateway STP, on page 195
• Configuring VXLAN EVPN Multihoming Traffic Flows, on page 199
• Configuring VLAN Consistency Checking, on page 211
• Configuring ESI ARP Suppression, on page 214

VXLAN EVPN Multihoming Overview

Introduction to Multihoming

Cisco Nexus platforms support vPC-based multihoming, where a pair of switches act as a single device for redundancy and both switches function in an active mode. With Cisco Nexus 31128PQ switch and 3100-V platform switches in VXLAN BGP EVPN environment, there are two solutions to support Layer 2 multihoming; the solutions are based on the Traditional vPC (emulated or virtual IP address) and the BGP EVPN techniques.

Traditional vPC utilizes a consistency check that is a mechanism used by the two switches that are configured as a vPC pair to exchange and verify their configuration compatibility. The BGP EVPN technique does not have the consistency check mechanism, but it uses LACP to detect the misconfigurations. It also eliminates the MCT link that is traditionally used by vPC and it offers more flexibility as each VTEP can be a part of one or more redundancy groups. It can potentially support many VTEPs in a given group.

BGP EVPN Multihoming Terminology

See this section for the terminology used in BGP EVPN multihoming:

- EVI: EVPN instance represented by the VNI.
- MAC-VRF: A container to house virtual forwarding table for MAC addresses. A unique route distinguisher and import/export target can be configured per MAC-VRF.
- ES: Ethernet Segment that can constitute a set of bundled links.
- ESI: Ethernet Segment Identifier to represent each ES uniquely across the network.

EVPN Multihoming Implementation

The EVPN overlay draft specifies adaptations to the BGP MPLS based EVPN solution to enable it to be applied as a network virtualization overlay with VXLAN encapsulation. The Provider Edge (PE) node role in BGP MPLS EVPN is equivalent to VTEP/Network Virtualization Edge device (NVE), where VTEPs use control plane learning and distribution via BGP for remote addresses instead of data plane learning.

There are 5 different route types currently defined:

- Ethernet Auto-Discovery (EAD) Route
- MAC advertisement Route
- Inclusive Multicast Route
- Ethernet Segment Route
- IP Prefix Route

BGP EVPN running on Cisco NX-OS uses route type-2 to advertise MAC and IP (host) information, route type-3 to carry VTEP information (specifically for ingress replication), and the EVPN route type-5 allows advertisements of IPv4 or IPv6 prefixes in an Network Layer Reachability Information (NLRI) with no MAC addresses in the route key.

With the introduction of EVPN multihoming, Cisco NX-OS software utilizes Ethernet Auto-discovery (EAD) route, where Ethernet Segment Identifier and the Ethernet Tag ID are considered to be part of the prefix in the NLRI. Since the end points reachability is learned via the BGP control plane, the network convergence time is a function of the number of MAC/IP routes that must be withdrawn by the VTEP in case of a failure scenario. To deal with such condition, each VTEP advertises a set of one or more Ethernet Auto-Discovery per ES routes for each locally attached Ethernet Segment and upon a failure condition to the attached segment, the VTEP withdraws the corresponding set of Ethernet Auto-Discovery per ES routes.

Ethernet Segment Route is the other route type that is being used by Cisco NX-OS software with EVPN multihoming, mainly for Designated Forwarder (DF) election for the BUM traffic. If the Ethernet Segment is multihomed, the presence of multiple DFs could result in forwarding the loops in addition to the potential packet duplication. Therefore, the Ethernet Segment Route (Type 4) is used to elect the Designated Forwarder and to apply Split Horizon Filtering. All VTEPs/PEs that are configured with an Ethernet Segment originate this route.

To summarize the new implementation concepts for the EVPN multihoming:

- EAD/ES: Ethernet Auto Discovery Route per ES that is also referred to as type-1 route. This route is used to converge the traffic faster during access failure scenarios. This route has Ethernet Tag of 0xFFFFFFFF.
- EAD/EVI: Ethernet Auto Discovery Route per EVI that is also referred to as type-1 route. This route is used for aliasing and load balancing when the traffic only hashes to one of the switches. This route cannot have Ethernet Tag value of 0xFFFFFF to differentiate it from the EAD/ES route.
- ES: Ethernet Segment route that is also referred to as type-4 route. This route is used for DF election for BUM traffic.
- Aliasing: It is used for load balancing the traffic to all the connected switches for a given Ethernet Segment using the type-1 EAD/EVI route. This is done irrespective of the switch where the hosts are actually learned.

- Mass Withdrawal: It is used for fast convergence during the access failure scenarios using the type-1 EAD/ES route.
- DF Election: It is used to prevent forwarding of the loops and the duplicates as only a single switch is allowed to decap and forward the traffic for a given Ethernet Segment.
- Split Horizon: It is used to prevent forwarding of the loops and the duplicates for the BUM traffic. Only the BUM traffic that originates from a remote site is allowed to be forwarded to a local site.

EVPN Multihoming Redundancy Group

Consider the dually homed topology, where switches L1 and L2 are distributed anycast VXLAN gateways that perform Integrated Routing and Bridging (IRB). Host H2 is connected to an access switch that is dually homed to both L1 and L2.

The access switch is connected to L1 and L2 via a bundled pair of physical links. The switch is not aware that the bundle is configured on two different devices on the other side. However, both L1 and L2 must be aware that they are a part of the same bundle.

Note that there is no Multichassis EtherChannel Trunk (MCT) link between L1 and L2 switches and each switch can have similar multiple bundle links that are shared with the same set of neighbors.

To make the switches L1 and L2 aware that they are a part of the same bundle link, the NX-OS software utilizes the Ethernet Segment Identifier (ESI) and the system MAC address (system-mac) that is configured under the interface (PO).

Ethernet Segment Identifier

EVPN introduces the concept of Ethernet Segment Identifier (ESI). Each switch is configured with a 10 byte ESI value under the bundled link that they share with the multihomed neighbor. The ESI value can be manually configured or auto-derived.

LACP Bundling

LACP can be turned ON for detecting ESI misconfigurations on the multihomed port channel bundle as LACP sends the ESI configured MAC address value to the access switch. LACP is not mandated along with ESI. A given ESI interface (PO) shares the same ESI ID across the VTEPs in the group.

The access switch receives the same configured MAC value from both switches (L1 and L2). Therefore, it puts the bundled link in the UP state. Since the ES MAC can be shared across all the Ethernet-segments on the switch, LACP PDUs use ES MAC as system MAC address and the admin_key carries the ES ID.

Cisco recommends running LACP between the switches and the access devices since LACP PDUs have a mechanism to detect and act on the misconfigured ES IDs. In case there is mismatch on the configured ES ID under the same PO, LACP brings down one of the links (first link that comes online stays up). By default, on most Cisco Nexus platforms, LACP sets a port to the suspended state if it does not receive an LACP PDU from the peer. This is based on the **lacp suspend-individual** command that is enabled by default. This command helps in preventing loops that are created due to the ESI configuration mismatch. Therefore, it is recommended to enable this command on the port-channels on the access switches and the servers.

In some scenarios (for example, POAP or NetBoot), it can cause the servers to fail to boot up because they require LACP to logically bring up the port. In case you are using static port channel and you have mismatched ES IDs, the MAC address gets learned from both L1 and L2 switches. Therefore, both the switches advertise

the same MAC address belonging to different ES IDs that triggers the MAC address move scenario. Eventually, no traffic is forwarded to that node for the MAC addresses that are learned on both L1 and L2 switches.

Guidelines and Limitations for VXLAN EVPN Multihoming

See the following limitations for configuring VXLAN EVPN multihoming:

- EVPN multihoming is supported on the Cisco Nexus 3100-V and 3132-Z platform switches only.
- Beginning with Cisco NX-OS Release 7.0(3)I7(1), ARP suppression is supported with EVPN multihoming.
- EVPN multihoming is supported with multihoming to two switches only.
- To enable EVPN multihoming, the spine switches should be running the minimum software version as Cisco NX-OS Release 7.0(3)I7(1) or later.
- Switchport trunk native VLAN is not supported on the trunk interfaces.
- Cisco recommends enabling LACP on ES PO.
- IPv6 is currently not supported.

Configuring VXLAN EVPN Multihoming

Enabling EVPN Multihoming

Cisco NX-OS allows either vPC based EVPN multihoming or ESI based EVPN multihoming. Both features should not be enabled together. ESI based multihoming is enabled using **evpn esi multihoming** CLI command. It is important to note that the command for ESI multihoming enables the Ethernet-segment configurations and the generation of Ethernet-segment routes on the switches.

The receipt of type-1 and type-2 routes with valid ESI and the path-list resolution are not tied to the **evpn esi multihoming** command. If the switch receives MAC/MAC-IP routes with valid ESI and the command is not enabled, the ES based path resolution logic still applies to these remote routes. This is required for interoperability between the vPC enabled switches and the ESI enabled switches.

Complete the following steps to configure EVPN multihoming:

Before you begin

VXLAN should be configured with BGP-EVPN before enabling EVPN ESI multihoming.

Procedure

	Command or Action	Purpose
Step 1	**evpn esi multihoming**	Enables EVPN multihoming globally.
Step 2	**ethernet-segment delay-restore time 30**	The ESI Port Channel remains down for 30 seconds after the core facing interfaces are up.
Step 3	**vlan-consistency-check**	Enables VLAN consistency check.

	Command or Action	Purpose

Step 4	**address-family l2vpn evpn maximum-paths <>maximum-paths ibgp <>**
	Example:
	`address-family l2vpn evpn`
	`maximum-paths 64`
	`maximum-paths ibgp 64`
		Enables BGP maximum-path to enable ECMP for the MAC routes. Otherwise, the MAC routes have only 1 VTEP as the next-hop. This configuration is needed under BGP in Global level.

Step 5	**evpn multihoming core-tracking**	Enables EVPN multihoming core-links. It tracks the uplink interfaces towards the core. If all uplinks are down, the local ES based the POs is shut down/suspended. This is mainly used to avoid black-holing South-to-North traffic when no uplinks are available.

Step 6	**interface port-channel Ethernet-segment <>System-mac <>**
	Example:
	`ethernet-segment 11`
	`system-mac 0000.0000.0011`
		Configures the local Ethernet Segment ID. The ES ID has to match on VTEPs where the PO is multihomed. The Ethernet Segment ID should be unique per PO.
		Configures the local system-mac ID that has to match on the VTEPs where the PO is multihomed. The system-mac address can be shared across multiple POs.

Step 7	**hardware access-list tcam region vpc-convergence 256** **Example:** `hardware access-list tcam region vpc-convergence 256`	Configures the TCAM. This command is used to configure the split horizon ACLs in the hardware. This command avoids BUM traffic duplication on the shared ES POs.

VXLAN EVPN Multihoming Configuration Examples

See the sample VXLAN EVPN multihoming configuration on the switches:

```
Switch 1 (L1)

evpn esi multihoming

ethernet-segment delay-restore time 180
vlan-consistency-check
router bgp 1001
   address-family l2vpn evpn
   maximum-paths ibgp 2

interface Ethernet2/1
  no switchport
  evpn multihoming core-tracking
  mtu 9216
  ip address 10.1.1.1/30
  ip pim sparse-mode
```

```
  no shutdown

interface Ethernet2/2
  no switchport
  evpn multihoming core-tracking
  mtu 9216
  ip address 10.1.1.5/30
  ip pim sparse-mode
  no shutdown

interface port-channel11
  switchport mode trunk
  switchport trunk allowed vlan 901-902,1001-1050
  ethernet-segment 2011
    system-mac 0000.0000.2011
  mtu 9216
```

Switch 2 (L2)

```
evpn esi multihoming

ethernet-segment delay-restore time 180
vlan-consistency-check
router bgp 1001
   address-family l2vpn evpn
   maximum-paths ibgp 2

interface Ethernet2/1
  no switchport
  evpn multihoming core-tracking
  mtu 9216
  ip address 10.1.1.2/30
  ip pim sparse-mode
  no shutdown

interface Ethernet2/2
  no switchport
  evpn multihoming core-tracking
  mtu 9216
  ip address 10.1.1.6/30
  ip pim sparse-mode
  no shutdown

interface port-channel11
  switchport mode trunk
  switchport access vlan 1001
  switchport trunk allowed vlan 901-902,1001-1050
  ethernet-segment 2011
    system-mac 0000.0000.2011
  mtu 9216
```

Configuring Layer 2 Gateway STP

Layer 2 Gateway STP Overview

Beginning with Cisco NX-OS Release 7.0(3)I7(1), EVPN multihoming is supported with the Layer 2 Gateway Spanning Tree Protocol (L2G-STP). The Layer 2 Gateway Spanning Tree Protocol (L2G-STP) builds a loop-free tree topology. However, the Spanning Tree Protocol root must always be in the VXLAN fabric. A bridge ID for the Spanning Tree Protocol consists of a MAC address and the bridge priority. When the system is running in the VXLAN fabric, the system automatically assigns the VTEPs with the MAC address c84c.75fa.6000 from a pool of reserved MAC addresses. As a result, each switch uses the same MAC address for the bridge ID emulating a single logical pseudo root.

The Layer 2 Gateway Spanning Tree Protocol (L2G-STP) is disabled by default on EVPN ESI multihoming VLANs. Use the **spanning-tree domain enable** CLI command to enable L2G-STP on all VTEPs. With L2G-STP enabled, the VXLAN fabric (all VTEPs) emulates a single pseudo root switch for the customer access switches. The L2G-STP is initiated to run on all VXLAN VLANs by default on boot up and the root is fixed on the overlay. With L2G-STP, the root-guard gets enabled by default on all the access ports. Use **spanning-tree domain** <*id*> to additionally enable Spanning Tree Topology Change Notification(STP-TCN), to be tunneled across the fabric.

All the access ports from VTEPs connecting to the customer access switches are in a *desg* forwarding state by default. All ports on the customer access switches connecting to VTEPs are either in root-port forwarding or alt-port blocking state. The root-guard kicks in if better or superior STP information is received from the customer access switches and it puts the ports in the *blk l2g_inc* state to secure the root on the overlay-fabric and to prevent a loop.

Guidelines for Moving to Layer 2 Gateway STP

Complete the following steps to move to Layer 2 gateway STP:

- With Layer 2 Gateway STP, root guard is enabled by default on all the access ports.
- With Layer 2 Gateway STP enabled, the VXLAN fabric (all VTEPs) emulates a single pseudo-root switch for the customer access switches.
- All access ports from VTEPs connecting to the customer access switches are in the **Desg FWD** state by default.
- All ports on customer access switches connecting to VTEPs are either in the root-port FWD or Altn BLK state.
- Root guard is activated if superior spanning-tree information is received from the customer access switches. This process puts the ports in **BLK L2GW_Inc** state to secure the root on the VXLAN fabric and prevent a loop.
- Explicit domain ID configuration is needed to enable spanning-tree BPDU tunneling across the fabric.
- As a best practice, you should configure all VTEPs with the lowest spanning-tree priority of all switches in the spanning-tree domain to which they are attached. By setting all the VTEPs as the root bridge, the entire VXLAN fabric appears to be one virtual bridge.

- ESI interfaces should not be enabled in spanning-tree edge mode to allow Layer 2 Gateway STP to run across the VTEP and access layer.
- You can continue to use ESIs or orphans (single-homed hosts) in spanning-tree edge mode if they directly connect to hosts or servers that do not run Spanning Tree Protocol and are end hosts.
- Configure all VTEPs that are connected by a common customer access layer in the same Layer 2 Gateway STP domain. Ideally, all VTEPs on the fabric on which the hosts reside and to which the hosts can move.
- The Layer 2 Gateway STP domain scope is global, and all ESIs on a given VTEP can participate in only one domain.
- Mappings between Multiple Spanning Tree (MST) instances and VLANs must be consistent across the VTEPs in a given Layer 2 Gateway STP domain.
- Non-Layer 2 Gateway STP enabled VTEPs cannot be directly connected to Layer 2 Gateway STP-enabled VTEPs. Performing this action results in conflicts and disputes because the non-Layer 2 Gateway STP VTEP keeps sending BPDUs and it can steer the root outside.
- Keep the current edge and the BPDU filter configurations on both the Cisco Nexus switches and the access switches after upgrading to the latest build.
- Enable Layer 2 Gateway STP on all the switches with a recommended priority and the *mst* instance mapping as needed. Use the commands **spanning-tree domain enable** and **spanning-tree mst <instance-id's> priority 8192**.
- Remove the BPDU filter configurations on the switch side first.
- Remove the BPDU filter configurations and the edge on the customer access switch.

  Now the topology converges with Layer 2 Gateway STP and any blocking of the redundant connections is pushed to the access switch layer.

Enabling Layer 2 Gateway STP on a Switch

Complete the following steps to enable Layer 2 Gateway STP on a switch.

Procedure

	Command or Action	Purpose
Step 1	**spanning-tree mode <rapid-pvst, mst>**	Enables Spanning Tree Protocol mode.
Step 2	**spanning-tree domain enable**	Enables Layer 2 Gateway STP on a switch. It disables Layer 2 Gateway STP on all EVPN ESI multihoming VLANs.
Step 3	**spanning-tree domain 1**	Explicit domain ID is needed to tunnel encoded BPDUs to the core and processes received from the core.
Step 4	**spanning-tree mst <id> priority 8192**	Configures Spanning Tree Protocol priority.
Step 5	**spanning-tree vlan <id> priority 8192**	Configures Spanning Tree Protocol priority.

	Command or Action	Purpose
Step 6	**spanning-tree domain disable**	Disables Layer 2 Gateway STP on a VTEP.

Example

All Layer 2 Gateway STP VLANs should be set to a lower spanning-tree priority than the customer-edge (CE) topology to help ensure that the VTEP is the spanning-tree root for this VLAN. If the access switches have a higher priority, you can set the Layer 2 Gateway STP priority to 0 to retain the Layer 2 Gateway STP root in the VXLAN fabric. See the following configuration example:

```
switch# show spanning-tree summary
Switch is in mst mode (IEEE Standard)
Root bridge for: MST0000
L2 Gateway STP bridge for: MST0000
L2 Gateway Domain ID: 1
Port Type Default                        is disable
Edge Port [PortFast] BPDU Guard Default  is disabled
Edge Port [PortFast] BPDU Filter Default is disabled
Bridge Assurance                         is enabled
Loopguard Default                        is disabled
Pathcost method used                     is long
PVST Simulation                          is enabled
STP-Lite                                 is disabled

Name                   Blocking Listening Learning Forwarding STP Active
---------------------- -------- --------- -------- ---------- ----------
MST0000                     0         0        0         12         12
---------------------- -------- --------- -------- ---------- ----------
1 mst                       0         0        0         12         12


switch# show spanning-tree vlan 1001

MST0000
  Spanning tree enabled protocol mstp

  Root ID    Priority    8192
             Address     c84c.75fa.6001   L2G-STP reserved mac+ domain id
             This bridge is the root
             Hello Time  2  sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    8192   (priority 8192 sys-id-ext 0)
             Address     c84c.75fa.6001
             Hello Time  2  sec  Max Age 20 sec  Forward Delay 15 sec
```

The output displays that the spanning-tree priority is set to 8192 (the default is 32768). Spanning-tree priority is set in multiples of 4096. The priority for individual instances is calculated as the priority and the Instance_ID. In this case, the priority is calculated as 8192 + 0 = 8192. With Layer 2 Gateway STP, access ports (VTEP ports connected to the access switches) have root guard enabled. If a superior BPDU is received on an edge port of a VTEP, the port is placed in the Layer 2 Gateway inconsistent state until the condition is cleared as displayed in the following example:

```
2016 Aug 29 19:14:19 TOR9-leaf4 %$ VDC-1 %$ %STP-2-L2GW_BACKBONE_BLOCK: L2 Gateway Backbone
 port inconsistency blocking port Ethernet1/1 on MST0000.
2016 Aug 29 19:14:19 TOR9-leaf4 %$ VDC-1 %$ %STP-2-L2GW_BACKBONE_BLOCK: L2 Gateway Backbone
 port inconsistency blocking port port-channel13 on MST0000.

switch# show spanning-tree

MST0000
  Spanning tree enabled protocol mstp
  Root ID    Priority    8192
             Address     c84c.75fa.6001
             This bridge is the root
             Hello Time  2  sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    8192   (priority 8192 sys-id-ext 0)
             Address     c84c.75fa.6001
             Hello Time  2  sec  Max Age 20 sec  Forward Delay 15 sec

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Po1              Desg FWD 20000     128.4096 Edge P2p
Po2              Desg FWD 20000     128.4097 Edge P2p
Po3              Desg FWD 20000     128.4098 Edge P2p
Po12            Desg BKN*2000       128.4107 P2p *L2GW_Inc
Po13            Desg BKN*1000       128.4108 P2p *L2GW_Inc
Eth1/1        Desg BKN*2000       128.1    P2p *L2GW_Inc
```

To disable Layer 2 Gateway STP on a VTEP, enter the **spanning-tree domain disable** CLI command. This command disables Layer 2 Gateway STP on all EVPN ESI multihomed VLANs. The bridge MAC address is restored to the system MAC address, and the VTEP may not necessarily be the root. In the following case, the access switch has assumed the root role because Layer 2 Gateway STP is disabled:

```
switch(config)# spanning-tree domain disable

switch# show spanning-tree summary
Switch is in mst mode (IEEE Standard)
Root bridge for: none
L2 Gateway STP                          is disabled
Port Type Default                       is disable
Edge Port [PortFast] BPDU Guard Default  is disabled
Edge Port [PortFast] BPDU Filter Default is disabled
Bridge Assurance                        is enabled
Loopguard Default                       is disabled
Pathcost method used                    is long
PVST Simulation                         is enabled
STP-Lite                                is disabled

Name                   Blocking Listening Learning Forwarding STP Active
---------------------- -------- --------- -------- ---------- ----------
MST0000                     4         0        0          8         12
---------------------- -------- --------- -------- ---------- ----------
1 mst                       4         0        0          8         12


switch# show spanning-tree vlan 1001

MST0000
  Spanning tree enabled protocol mstp
  Root ID    Priority    4096
             Address     00c8.8ba6.5073
```

```
            Cost         0
            Port         4108 (port-channel13)
            Hello Time   2  sec  Max Age 20 sec  Forward Delay 15 sec

Bridge ID   Priority     8192   (priority 8192 sys-id-ext 0)
            Address      5897.bd1d.db95
            Hello Time   2  sec  Max Age 20 sec  Forward Delay 15 sec
```

With Layer 2 Gateway STP, the access ports on VTEPs cannot be in an edge port, because they behave like normal spanning-tree ports, receiving BPDUs from the access switches. In that case, the access ports on VTEPs lose the advantage of rapid transmission, instead forwarding on Ethernet segment link flap. (They have to go through a proposal and agreement handshake before assuming the FWD-Desg role).

Configuring VXLAN EVPN Multihoming Traffic Flows

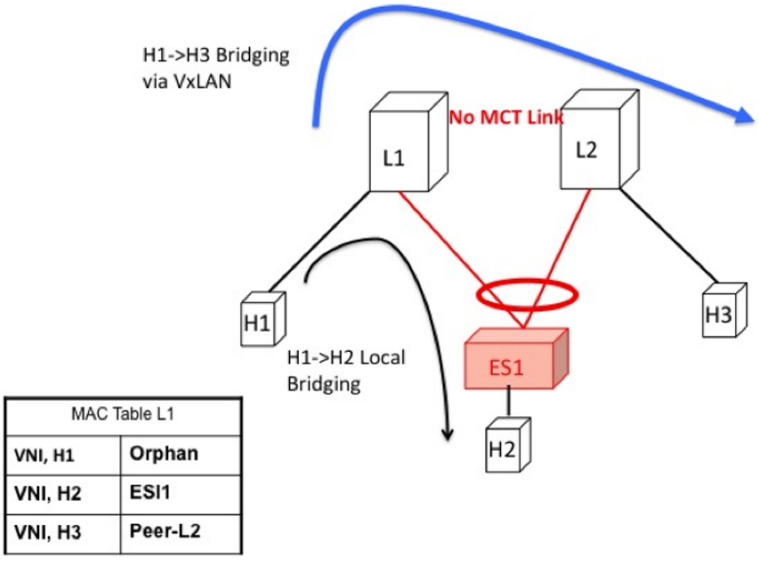
EVPN Multihoming Local Traffic Flows

All switches that are a part of the same redundancy group (as defined by the ESI) act as a single virtual switch with respect to the access switch/host. However, there is no MCT link present to bridge and route the traffic for local access.

Locally Bridged Traffic

Host H2 is dually homed whereas hosts H1 and H3 are single-homed (also known as orphans). The traffic is bridged locally from H1 to H2 via L1. However, if the packet needs to be bridged between the orphans H1 and H3, the packet must be bridged via the VXLAN overlay.

Figure 14: Local Bridging at L1. H1->H3 bridging via VXLAN. In vPC, H1->H3 will be via MCT link.

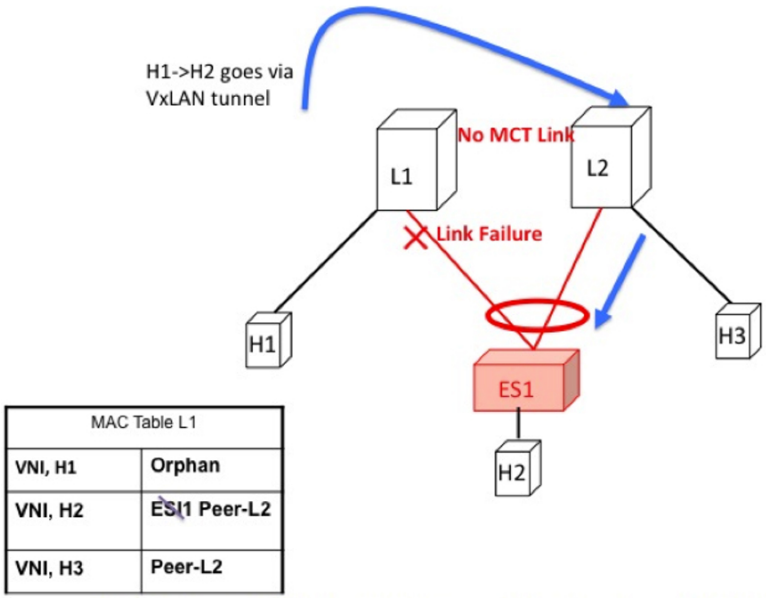


Access Failure for Locally Bridged Traffic

If the ESI link at L1 fails, there is no path for the bridged traffic to reach from H1 to H2 except via the overlay. Therefore, the local bridged traffic takes the sub-optimal path, similar to the H1 to H3 orphan flow.

Note When such condition occurs, the MAC table entry for H2 changes from a local route pointing to a port channel interface to a remote overlay route pointing to peer-ID of L2. The change gets percolated in the system from BGP.

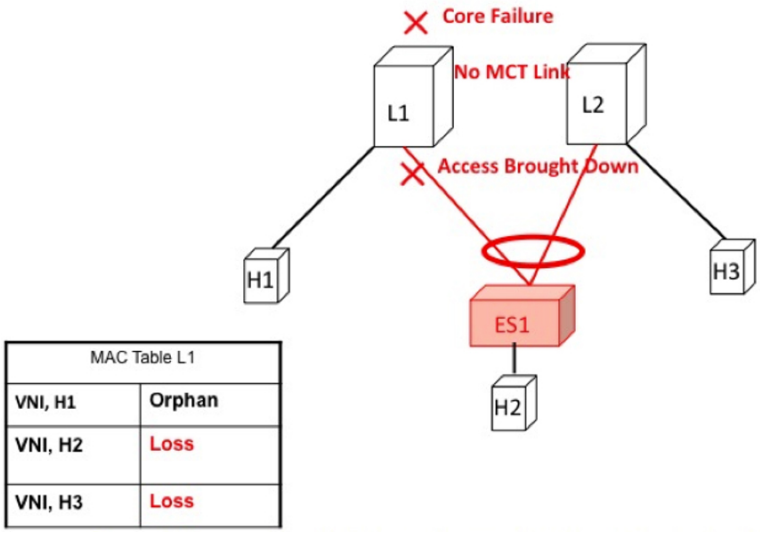
Figure 15: ES1 failure on L1. H1->H2 is now bridged over VXLAN tunnel.



Core Failure for Locally Bridged Traffic

If switch L1 gets isolated from the core, it must not continue to attract access traffic, as it will not be able to encapsulate and send it on the overlay. This means that the access links must be brought down at L1 if L1 loses core reachability. In this scenario, orphan H1 loses all connectivity to both remote and locally attached hosts since there is no dedicated MCT link.

Figure 16: Core failure on L1. H1->H2 loses all connectivity as there is no MCT.



Locally Routed Traffic

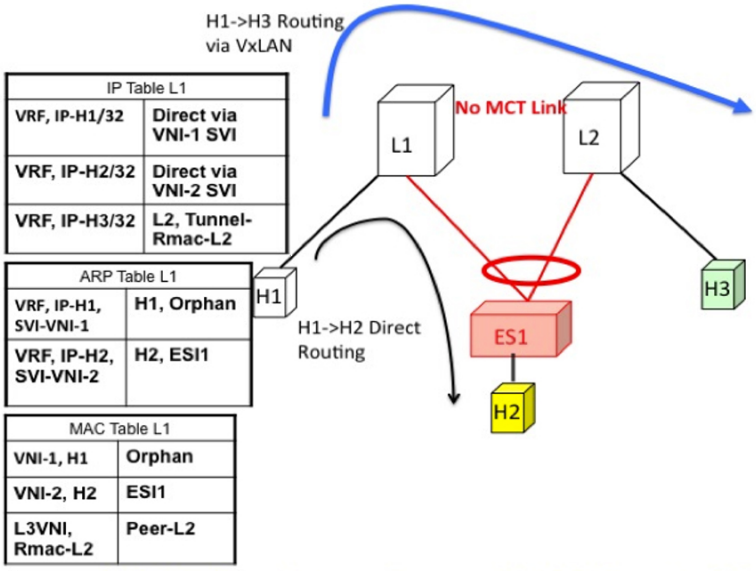
Consider H1, H2, and H3 being in different subnets and L1/L2 being distributed anycast gateways.

Any packet that is routed from H1 to H2 is directly sent from L1 via native routing.

However, host H3 is not a locally attached adjacency, unlike in vPC case where the ARP entry syncs to L1 as a locally attached adjacency. Instead, H3 shows up as a remote host in the IP table at L1, installed in the context of L3 VNI. This packet must be encapsulated in the router-MAC of L2 and routed to L2 via VXLAN overlay.

Therefore, routed traffic from H1 to H3 takes place exactly in the same fashion as routed traffic between truly remote hosts in different subnets.

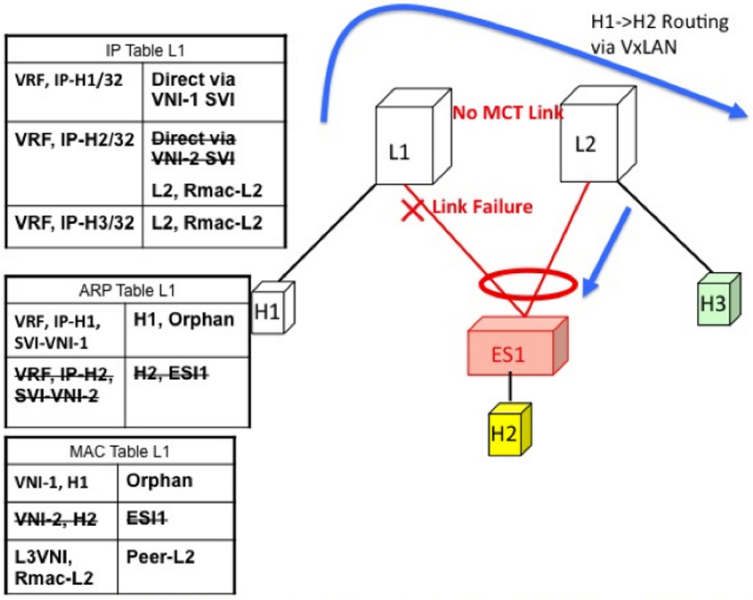
Figure 17: L1 is Distributed Anycast Gateway. H1, H2, and H3 are in different VLANs. H1->H3 routing happens via VXLAN tunnel encapsulation. In VPC, H3 ARP would have been synced via MCT and direct routing.



Access Failure for Locally Routed Traffic

In case the ESI link at switch L1 fails, there is no path for the routed traffic to reach from H1 to H2 except via the overlay. Therefore, the local routed traffic takes the sub-optimal path, similar to the H1 to H3 orphan flow.

Figure 18: H1, H2, and H3 are in different VLANs. ESI fails on L1. H1->H2 routing happens via VXLAN tunnel encapsulation.

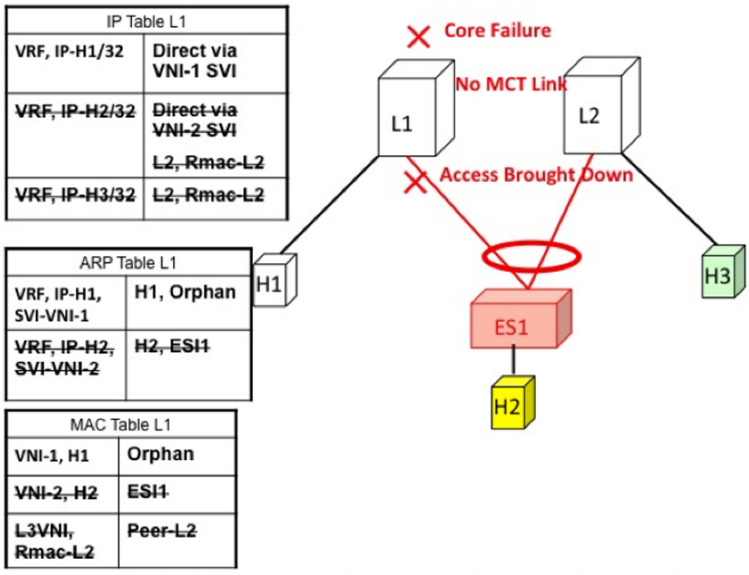


Core Failure for Locally Routed Traffic

If switch L1 gets isolated from the core, it must not continue to attract access traffic, as it will not be able to encapsulate and send it on the overlay. It means that the access links must be brought down at L1 if L1 loses core reachability.

In this scenario, orphan H1 loses all connectivity to both remote and locally attached hosts as there is no dedicated MCT link.

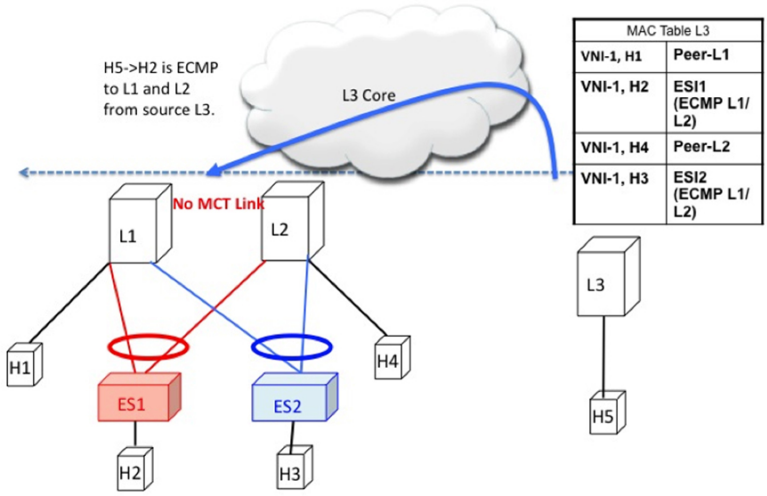
Figure 19: H1, H2, and H3 are in different VLANs. Core fails on L1. Access is brought down. H1 loses all connectivity.



EVPN Multihoming Remote Traffic Flows

Consider a remote switch L3 that sends bridged and routed traffic to the multihomed complex comprising of switches L1 and L2. As there is no virtual or emulated IP representing this MH complex, L3 must do ECMP at the source for both bridged and routed traffic. This section describes how the ECMP is achieved at switch L3 for both bridged and routed cases and how the system interacts with core and access failures.

Figure 20: Layer 2 VXLAN Gateway. L3 performs MAC ECMP to L1/L2.



Remote Bridged Traffic

Consider a remote host H5 that wants to bridge traffic to host H2 that is positioned behind the EVPN MH Complex (L1, L2). Host H2 builds an ECMP list in accordance to the rules defined in RFC 7432. The MAC table at switch L3displays that the MAC entry for H2 points to an ECMP PathList comprising of IP-L1 and IP-L2. Any bridged traffic going from H5 to H2 is VXLAN encapsulated and load balanced to switches L1 and L2. When making the ECMP list, the following constructs need to be kept in mind:

- Mass Withdrawal: Failures causing PathList correction should be independent of the scale of MACs.
- Aliasing: PathList Insertions may be independent of the scale of MACs (based on support of optional routes).

Below are the main constructs needed to create this MAC ECMP PathList:

Ethernet Auto Discovery Route (Type 1) per ES

EVPN defines a mechanism to efficiently and quickly signal the need to update their forwarding tables upon the occurrence of a failure in connectivity to an Ethernet Segment. Having each PE advertise a set of one or more Ethernet A-D per ES route for each locally attached Ethernet Segment does this.

Ethernet Auto Discovery Route (Route Type 1) per ES		
NLRI	Route Type	Ethernet Segment (Type 1)
	Route Distinguisher	Router-ID: Segment-ID (VNID << 8)
	ESI	<Type: 1B><MAC: 6B><LD: 3B>
	Ethernet Tag	MAX-ET
	MPLS Label	0

Ethernet Auto Discovery Route (Route Type 1) per ES		
ATTRS	ESI Label Extended Community ESI Label = 0	Single Active = False
	Next-Hop	NVE Loopback IP
	Route Target	Subset of List of RTs of MAC-VRFs associated to all the EVIs active on the ES

MAC-IP Route (Type 2)

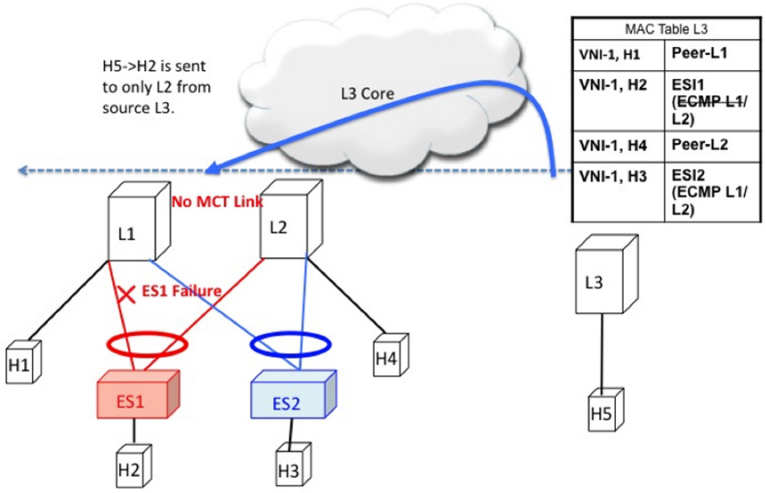
MAC-IP Route remains the same as used in the current vPC multihoming and standalone single-homing solutions. However, now it has a non-zero ESI field that indicates that this is a multihomed host and it is a candidate for ECMP Path Resolution.

MAC IP Route (Route Type 2)		
NLRI	Route Type	MAC IP Route (Type 2)
	Route Distinguisher	RD of MAC-VRF associated to the Host
	ESI	<Type : 1B><MAC : 6B><LD : 3B>
	Ethernet Tag	MAX-ET
	MAC Addr	MAC Address of the Host
	IP Addr	IP Address of the Host
	Labels	L2VNI associated to the MAC-VRF L3VNI associated to the L3-VRF
ATTRS	Next-Hop	Loopback of NVE
	RT Export	RT configured under MAC-VRF (AND/OR) L3-VRF associated to the host

Access Failure for Remote Bridged Traffic

In the condition of a failure of ESI links, it results in mass withdrawal. The EAD/ES route is withdrawn leading the remote device to remote the switch from the ECMP list for the given ES.

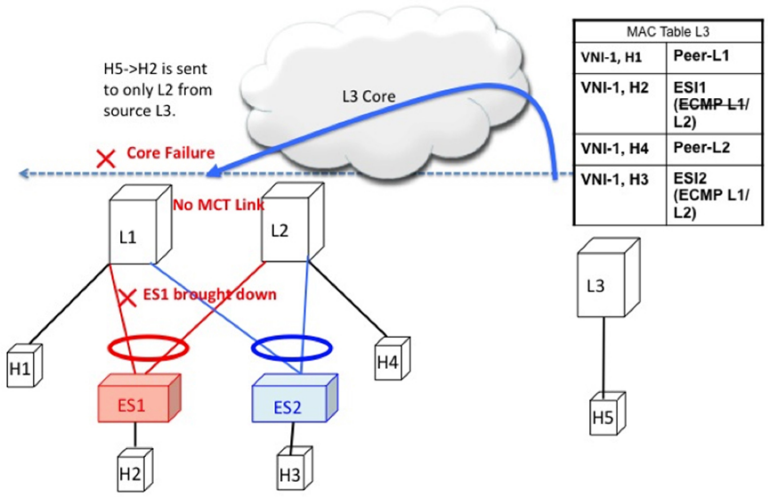
Figure 21: Layer 2 VXLAN Gateway. ESI failure on L1. L3 withdraws L1 from MAC ECMP list. This will happen due to EAD/ES mass withdrawal from L1.



Core Failure for Remote Bridged Traffic

If switch L1 gets isolated from the core, it must not continue to attract access traffic, as it is not able to encapsulate and send it on the overlay. It means that the access links must be brought down at L1 if L1 loses core reachability.

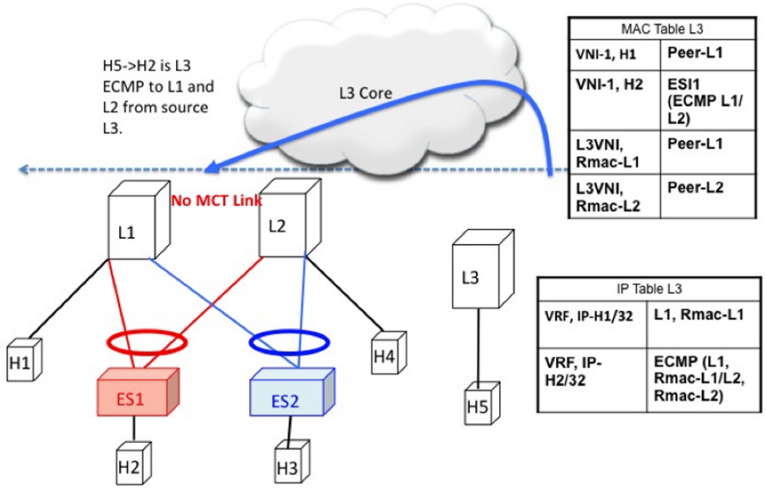
Figure 22: Layer 2 VXLAN Gateway. Core failure at L1. L3 withdraws L1 from MAC ECMP list. This will happen due to route reachability to L1 going away at L3.



Remote Routed Traffic

Consider L3 being a Layer 3 VXLAN Gateway and H5 and H2 belonging to different subnets. In that case, any inter-subnet traffic going from L3 to L1/L2 is routed at L3, that is a distributed anycast gateway. Both L1 and L2 advertise the MAC-IP route for Host H2. Due to the receipt of these routes, L3 builds an L3 ECMP list comprising of L1 and L2.

Figure 23: Layer 3 VXLAN Gateway. L3 does IP ECMP to L1/L2 for inter subnet traffic.



Access Failure for Remote Routed Traffic

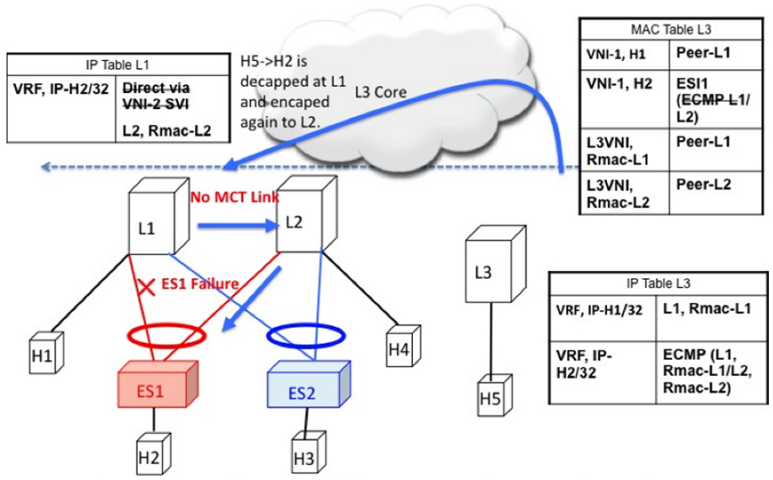
If the access link pointing to ES1 goes down on L1, the mass withdrawal route is sent in the form of EAD/ES and that causes L3 to remove L1 from the MAC ECMP PathList, leading the intra-subnet (L2) traffic to converge quickly. L1 now treats H2 as a remote route reachable via VxLAN Overlay as it is no longer directly connected through the ESI link. This causes the traffic destined to H2 to take the suboptimal path L3->L1->L2.

Inter-Subnet traffic H5->H2 will follow the following path:

- Packet are sent by H5 to gateway at L3.
- L3 performs symmetric IRB and routes the packet to L1 via VXLAN overlay.
- L1 decaps the packet and performs inner IP lookup for H2.
- H2 is a remote route. Therefore, L1 routes the packet to L2 via VXLAN overlay.
- L2 decaps the packet and performs an IP lookup and routes it to directly attached SVI.

Hence the routing happens 3 times, once each at L3, L1, and L2. This sub-optimal behavior continues until Type-2 route is withdrawn by L1 by BGP.

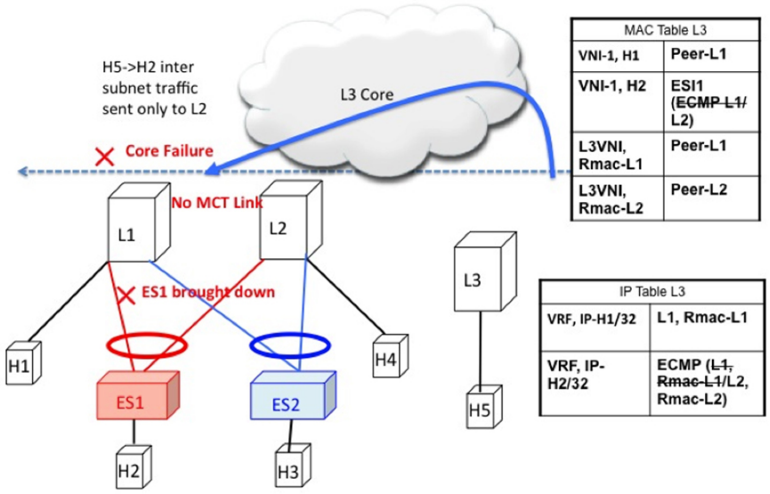
Figure 24: Layer 3 VXLAN Gateway. ESI failure causes ES mass withdrawal that only impacts L2 ECMP. L3 ECMP continues until Type2 is withdrawn. L3 traffic reaches H2 via suboptimal path L3->L1->L2 until then.



Core Failure for Remote Routed Traffic

Core Failure for Remote Routed Traffic behaves the same as core failure for remote bridged traffic. As the underlay routing protocol withdraws L1's loopback reachability from all remote switches, L1 is removed from both MAC ECMP and IP ECMP lists everywhere.

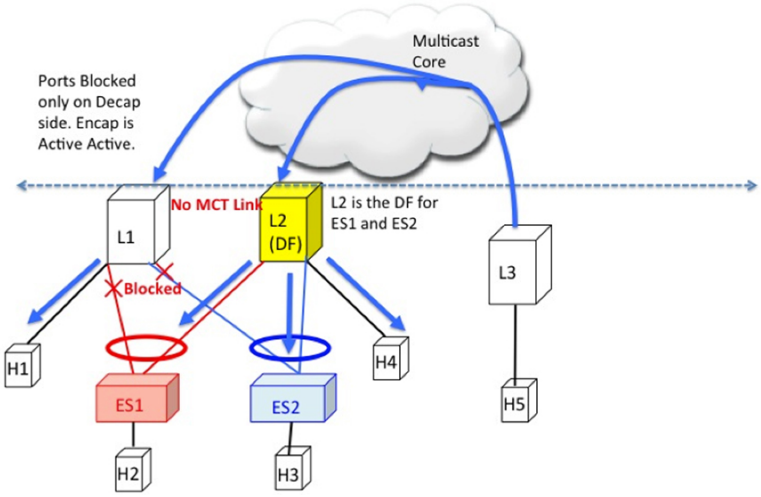
Figure 25: Layer 3 VXLAN Gateway. Core failure. All L3 ECMP paths to L1 are withdrawn at L3 due to route reachability going away.



EVPN Multihoming BUM Flows

NX-OS supports multicast core in the underlay with ESI. Consider BUM traffic originating from H5. The BUM packets are encapsulated in the multicast group mapped to the VNI. Because both L1 and L2 have joined the shared tree (*, G) for the underlay group based on the L2VNI mapping, both receive a copy of the BUM traffic.

Figure 26: BUM traffic originating at L3. L2 is the DF for ES1 and ES2. L2 decapsulates and forwards to ES1, ES2 and orphan. L1 decapsulates and only forwards to orphan.



Designated Forwarder

It is important that only one of the switches in the redundancy group decaps and forwards BUM traffic over the ESI links. For this purpose, a unique Designated Forwarder (DF) is elected on a per Ethernet Segment basis. The role of the DF is to decap and forward BUM traffic originating from the remote segments to the destination local segment for which the device is the DF. The main aspects of DF election are:

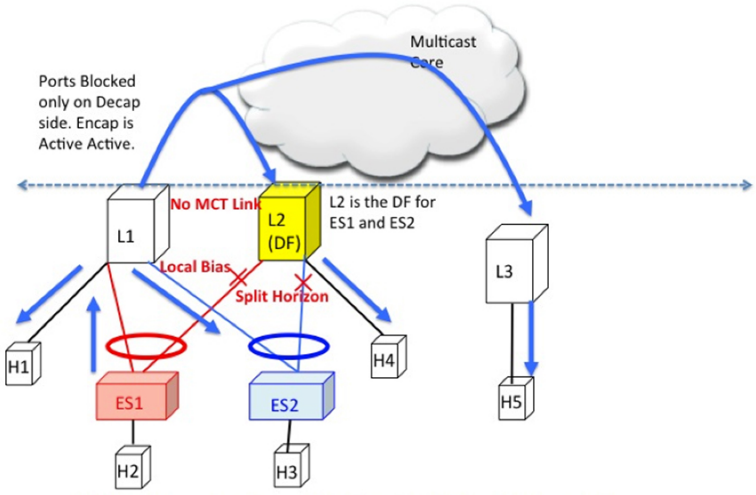
- DF Election is per (ES, VLAN) basis. There can be a different DF for ES1 and ES2 for a given VLAN.
- DF election result only applies to BUM traffic on the RX side for decap.
- Every switch must decap BUM traffic to forward it to singly homed or orphan links.
- Duplication of DF role leads to duplicate packets or loops in a DHN. Therefore, there must be a unique DF on per (ES, VLAN) basis.

Split Horizon and Local Bias

Consider BUM traffic originating from H2. Consider that this traffic is hashed at L1. L1 encapsulates this traffic in Overlay Multicast Group and sends the packet out to the core. All switches that have joined this multicast group with same L2VNI receive this packet. Additionally, L1 also locally replicates the BUM packet on all directly connected orphan and ESI ports. For example, if the BUM packet originated from ES1, L1 locally replicates it to ES2 and the orphan ports. This technique to replicate to all the locally attached links is termed as local-bias.

Remote switches decap and forward it to their ESI and orphan links based on the DF state. However, this packet is also received at L2 that belongs to the same redundancy group as the originating switch L1. L2 must decap the packet to send it to orphan ports. However, even through L2 is the DF for ES1, L2 must not forward this packet to ES1 link. This packet was received from a peer that shares ES1 with L1 as L1 would have done local-bias and duplicate copies should not be received on ES2. Therefore L2 (DF) applies a split-horizon filter for L1-IP on ES1 and ES2 that it shares with L1. This filter is applied in the context of a VLAN.

Figure 27: BUM traffic originating at L1. L2 is the DF for ES1 and ES2. However, L2 must perform split horizon check here as it shares ES1 and ES2 with L1. L2 however



Ethernet Segment Route (Type 4)

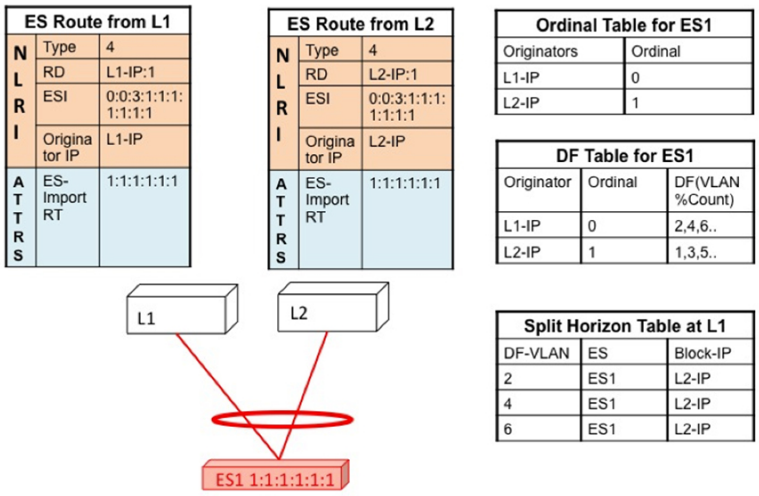
The Ethernet Segment Route is used to elect the Designated Forwarder and to apply Split Horizon Filtering. All the switches that are configured with an Ethernet Segment originate from this route. Ethernet Segment Route is exported and imported when ESI is locally configured under the PC.

Ethernet Segment Route (Route Type 4)		
NLRI	Route Type	Ethernet Segment (Type 4)
	RD	Router-ID: Base + Port Channel Number
	ESI	<Type : 1B><MAC : 6B><LD : 3B>
	Originator IP	NVE loopback IP
ATTRS	ES-Import RT	6 Byte MAC derived from ESI

DF Election and VLAN Carving

Upon configuration of the ESI, both L1 and L2 advertises the ES route. The ESI MAC is common between L1 and L2 and unique in the network. Therefore, only L1 and L2 import each other's ES routes.

Figure 28: If VLAN % count equals to ordinal, take up DF role.



Core and Site Failures for BUM Traffic

If the access link pertaining to ES1 fails at L1, L1 withdraws the ES route for ES1. This leads to a change triggering re-compute the DF. Since L2 is the only TOR left in the Ordinal Table, it takes over DF role for all VLANs.

BGP EVPN multihoming on Cisco Nexus 3100 Series switches provides minimum operational and cabling expenditure, provisioning simplicity, flow based load balancing, multi pathing, and fail-safe redundancy.

Configuring VLAN Consistency Checking

Overview of VLAN Consistency Checking

In a typical multihoming deployment scenario, host 1 belonging to VLAN X sends traffic to the access switch and then the access switch sends the traffic to both the uplinks towards VTEP1 and VTEP2. The access switch does not have the information about VLAN X configuration on VTEP1 and VTEP2. VLAN X configuration mismatch on VTEP1 or VTEP2 results in a partial traffic loss for host 1. VLAN consistency checking helps to detect such configuration mismatch.

For VLAN consistency checking, CFSoIP is used. Cisco Fabric Services (CFS) provides a common infrastructure to exchange the data across the switches in the same network. CFS has the ability to discover CFS capable switches in the network and to discover the feature capabilities in all the CFS capable switches. You can use CFS over IP (CFSoIP) to distribute and synchronize a configuration on one Cisco device or with all other Cisco devices in your network.

CFSoIP uses multicast to discover all the peers in the management IP network. For EVPN multihoming VLAN consistency checking, it is recommended to override the default CFS multicast address with the **cfs ipv4 mcast-address** <*mcast address*> CLI command. To enable CFSoIP, the **cfs ipv4 distribute** CLI command should be used.

When a trigger (for example, device booting up, VLAN configuration change, VLANs administrative state change on the ethernet-segment port-channel) is issued on one of the multihoming peers, a broadcast request

with a snapshot of configured and administratively up VLANs for the ethernet-segment (ES) is sent to all the CFS peers.

When a broadcast request is received, all CFS peers sharing the same ES as the requestor respond with their VLAN list (configured and administratively up VLAN list per ES). The VLAN consistency checking is run upon receiving a broadcast request or a response.

A 15 seconds timer is kicked off before sending a broadcast request. On receiving the broadcast request or response, the local VLAN list is compared with that of the ES peer. The VLANs that do not match are suspended. Newly matched VLANs are no longer suspended.

VLAN consistency checking runs for the following events:

- Global VLAN configuration: Add, delete, shut, or no shut events.

  Port channel VLAN configuration: Trunk allowed VLANs added or removed or access VLAN changed.

- CFS events: CFS peer added or deleted or CFSoIP configuration is removed.

- ES Peer Events: ES peer added or deleted.

The broadcast request is retransmitted if a response is not received. VLAN consistency checking fails to run if a response is not received after 3 retransmissions.

VLAN Consistency Checking Guidelines and Limitations

See the following guidelines and limitations for VLAN consistency checking:

- The VLAN consistency checking uses CFSoIP. Out-of-band access through a management interface is mandatory on all multihoming switches in the network.
- It is recommended to override the default CFS multicast address with the CLI **cfs ipv4 mcast-address** <*mcast address*> command.
- The VLAN consistency check cannot detect a mismatch in **switchport trunk native vlan** configuration.
- CFSoIP and CFSoE should not be used in the same device.
- CFSoIP should not be used in devices that are not used for VLAN consistency checking.
- If CFSoIP is required in devices that do not participate in VLAN consistency checking, a different multicast group should be configured for devices that participate in VLAN consistency with the CLI **cfs ipv4 mcast-address** <*mcast address*> command.

Configuring VLAN Consistency Checking

Displaying Show command Output for VLAN Consistency Checking

See the following show commands output for VLAN consistency checking.

To list the CFS peers, use the **sh cfs peers name nve** CLI command.

```
switch# sh cfs peers name nve

Scope      : Physical-ip
```

```
------------------------------------------------------------------------
 Switch WWN              IP Address
------------------------------------------------------------------------
 20:00:f8:c2:88:23:19:47 172.31.202.228                        [Local]
                         Switch
 20:00:f8:c2:88:90:c6:21 172.31.201.172                        [Not Merged]
 20:00:f8:c2:88:23:22:8f 172.31.203.38                         [Not Merged]
 20:00:f8:c2:88:23:1d:e1 172.31.150.132                        [Not Merged]
 20:00:f8:c2:88:23:1b:37 172.31.202.233                        [Not Merged]
 20:00:f8:c2:88:23:05:1d 172.31.150.134                        [Not Merged]
```

The **show nve ethernet-segment** command now displays the following details:

- The list of VLANs for which consistency check is failed.
- Remaining value (in seconds) of the global VLAN CC timer.

```
switch# sh nve ethernet-segment
ESI Database
---------------------------------------
ESI: 03aa.aaaa.aaaa.aa00.0001,
   Parent interface: port-channel2,
  ES State: Up
  Port-channel state: Up
  NVE Interface: nve1
   NVE State: Up
   Host Learning Mode: control-plane
  Active Vlans: 3001-3002
   DF Vlans: 3002
   Active VNIs: 30001-30002
  CC failed VLANs: 0-3000,3003-4095
  CC timer status: 10 seconds left
  Number of ES  members: 2
  My ordinal: 0
  DF timer start time: 00:00:00
  Config State: config-applied
  DF List: 201.1.1.1 202.1.1.1
  ES route added to L2RIB: True
  EAD routes added to L2RIB: True
```

See the following Syslog output:

```
switch(config)# 2017 Jan ?7 19:44:35 Switch %ETHPORT-3-IF_ERROR_VLANS_SUSPENDED: VLANs
2999-3000 on Interface port-channel40 are being suspended.
(Reason: SUCCESS)

After Fixing configuration
2017 Jan ?7 19:50:55 Switch %ETHPORT-3-IF_ERROR_VLANS_REMOVED: VLANs 2999-3000 on Interface
 port-channel40 are removed from suspended state.
```

Configuring ESI ARP Suppression

Overview of ESI ARP Suppression

ESI ARP suppression is an extension of already available ARP suppression solution in VXLAN-EVPN. This feature is supported on top of ESI multihoming solution, that is on top of VXLAN-EVPN solution. ARP suppression is an optimization on top of BGP-EVPN multihoming solution. ARP broadcast is one of the most significant part of broadcast traffic in data centers. ARP suppression significantly cuts down on ARP broadcast in the data center.

ARP request from host is normally flooded in the VLAN. You can optimize flooding by maintaining an ARP cache locally on the access switch. ARP cache is maintained by the ARP module. ARP cache is populated by snooping all the ARP packets from the access or server side. Initial ARP requests are broadcasted to all the sites. Subsequent ARP requests are suppressed at the first hop leaf and they are answered locally. In this way, the ARP traffic across overlay can be significantly reduced.

ARP suppression is only supported with BGP-EVPN (distributed gateway).

ESI ARP suppression is a per-VNI (L2-VNI) feature. ESI ARP suppression is supported in both L2 (no SVI) and L3 modes. Beginning with Cisco NX-OS Release 7.0(3)I7(1), only L3 mode is supported.

The ESI ARP suppression cache is built by:

- Snooping all ARP packets and populating ARP cache with the source IP and MAC bindings from the request.
- Learning IP-host or MAC-address information through BGP EVPN MAC-IP route advertisement.

Upon receiving the ARP request, the local cache is checked to see if the response can be locally generated. If the cache lookup fails, the ARP request can be flooded. This helps with the detection of the silent hosts.

Limitations for ESI ARP Suppression

See the following limitations for ESI ARP suppression:

- ESI multihoming solution is supported only on Cisco Nexus 3100 platform switches.
- ESI ARP suppression is only supported in L3 (SVI) mode.
- ESI ARP suppression cache limit is 64K that includes both local and remote entries.

Configuring ESI ARP Suppression

For ARP suppression VACLs to work, configure the TCAM carving using the **hardware access-list tcam region arp-ether 256** CLI command.

```
Interface nve1
   no shutdown
   source-interface loopback1
   host-reachability protocol bgp
   member vni 10000
```

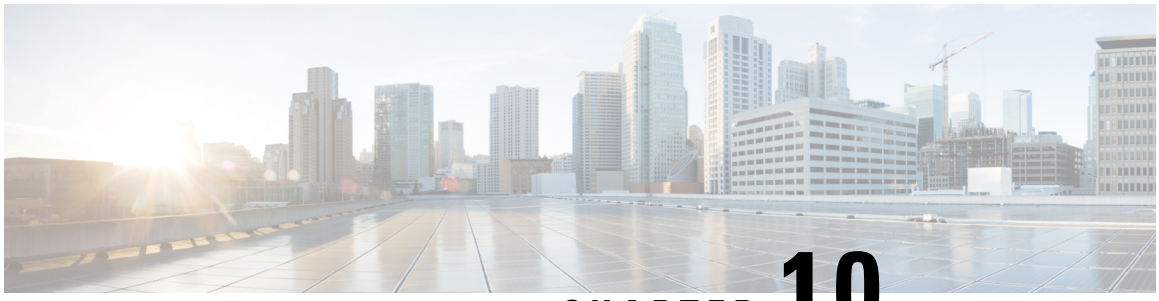
```
      suppress-arp
  mcast-group 224.1.1.10
```

Displaying Show Commands for ESI ARP Suppression

See the following Show commands output for ESI ARP suppression:

```
switch# show ip arp suppression-cache ?
 detail        Show details
   local          Show local entries
   remote     Show remote entries
   statistics  Show statistics
   summary Show summary
   vlan           L2vlan

switch# show ip arp suppression-cache local

Flags: + - Adjacencies synced via CFSoE
       L - Local Adjacency
       R - Remote Adjacency
       L2 - Learnt over L2 interface
      PS - Added via L2RIB, Peer Sync
      RO - Dervied from L2RIB Peer Sync Entry

Ip Address      Age      Mac Address    Vlan Physical-ifindex    Flags              Remote
 Vtep Addrs

61.1.1.20       00:07:54 0000.0610.0020  610 port-channel20     L
61.1.1.30       00:07:54 0000.0610.0030  610 port-channel2       L[PS RO]
61.1.1.10       00:07:54 0000.0610.0010  610 Ethernet1/96        L


switch# show ip arp suppression-cache remote
 Flags: + - Adjacencies synced via CFSoE
        L - Local Adjacency
        R - Remote Adjacency
        L2 - Learnt over L2 interface
          PS - Added via L2RIB, Peer Sync
          RO - Dervied from L2RIB Peer Sync Entry
          Ip Address      Age      Mac Address         Vlan    Physical-ifindex    Flags
    Remote Vtep Addrs
 61.1.1.40       00:48:37 0000.0610.0040   610     (null)                      R
 VTEP1, VTEP2.. VTEPn


switch# show ip arp suppression-cache detail
       Flags: + - Adjacencies synced via CFSoE
       L - Local Adjacency
       R - Remote Adjacency
       L2 - Learnt over L2 interface
       PS - Added via L2RIB, Peer Sync
       RO - Derived from L2RIB Peer Sync Entry

             Ip Address      Age      Mac Address    Vlan Physical-ifindex    Flags
     Remote Vtep Addrs
 61.1.1.20       00:00:07 0000.0610.0020  610 port-channel20     L
 61.1.1.30       00:00:07 0000.0610.0030  610 port-channel2       L[PS RO]
 61.1.1.10       00:00:07 0000.0610.0010  610 Ethernet1/96        L
 61.1.1.40       00:00:07 0000.0610.0040  610 (null)                    R
 VTEP1, VTEP2.. VTEPn
```

```
switch# show ip arp suppression-cache summary
 IP ARP suppression-cache Summary
 Remote          :1
 Local              :3
 Total               :4
switch# show ip arp suppression-cache statistics
 ARP packet statistics for suppression-cache
 Suppressed:
 Total 0, Requests 0, Requests on L2 0, Gratuitous 0, Gratuitous on L2 0
 Forwarded :
 Total: 364
  L3 mode :      Requests 364, Replies 0
  Request on core port 364, Reply on core port 0
                 Dropped 0
  L2 mode :      Requests 0, Replies 0
                 Request on core port 0, Reply on core port 0
                 Dropped 0
 Received:
 Total: 3016
 L3 mode:       Requests 376, Replies 2640
  Local Request 12, Local Responses 2640
                 Gratuitous 0, Dropped 0
  L2 mode :      Requests 0, Replies 0
                 Gratuitous 0, Dropped 0


switch# sh ip arp multihoming-statistics vrf all
ARP Multihoming statistics for all contexts
Route Stats
============
 Receieved ADD from L2RIB        :1756 | 1756:Processed ADD from L2RIB Receieved DEL from
L2RIB        :88 | 87:Processed DEL from L2RIB Receieved PC shut from L2RIB    :0 |
1755:Processed PC shut from L2RIB Receieved remote UPD from L2RIB :5004 | 0:Processed remote
 UPD from L2RIB
ERRORS
=======
Multihoming ADD error invalid flag         :0
Multihoming DEL error invalid flag         :0
Multihoming ADD error invalid current state:0
Multihoming DEL error invalid current state:0
Peer sync DEL error MAC mismatch           :0
Peer sync DEL error second delete          :0
Peer sync DEL error deleteing TL route     :0
True local DEL error deleteing PS RO route :0

switch#
```



CHAPTER 10

IPv6 Across a VXLAN EVPN Fabric

- Overview of IPv6 Across a VXLAN EVPN Fabric, on page 217
- Configuring IPv6 Across a VXLAN EVPN Fabric Example, on page 217
- Show Command Examples, on page 220

Overview of IPv6 Across a VXLAN EVPN Fabric

This section provides an example configuration that enables IPv6 in the overlay of a VXLAN EVPN fabric.

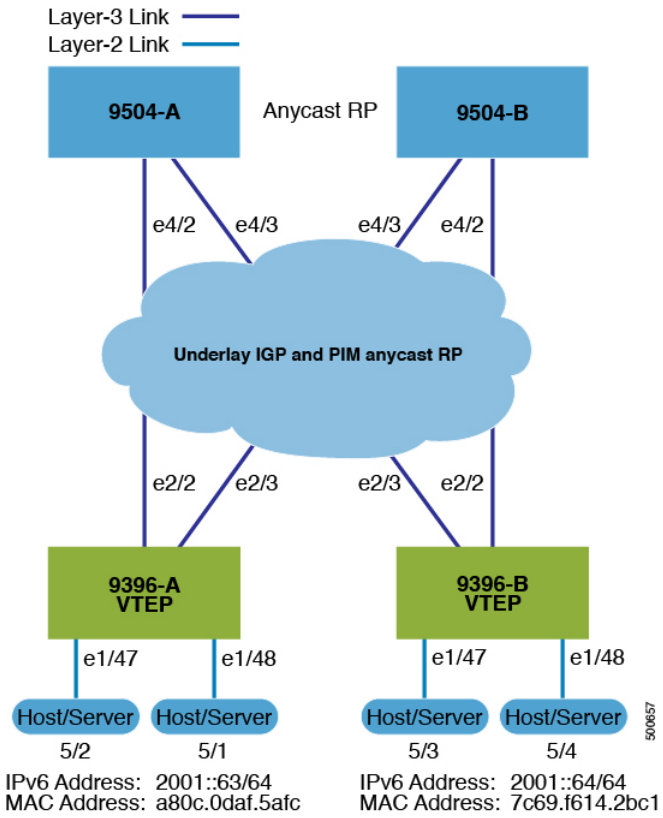
The VXLAN encapsulation mechanism encapsulates the IPv6 packets in the overlay as IPv4 UDP packets and uses IPv4 routing to transport the VXLAN encapsulated traffic.

To enable IPv6 across a VXLAN EVPN fabric, the IPv6 address family is included in VRF, BGP, and EVPN. IPv6 routes are initiated in the tenant VRF IPv6 unicast address-family on a VTEP and are advertised in the VXLAN fabric through the L2VPN EVPN address family as EVPN route-type 2 or 5.

Note These routes are advertised as EVPN routes on the SPINE.

Configuring IPv6 Across a VXLAN EVPN Fabric Example

Topology for the example:



Note In the example:

- Configuration for hosts in VLAN 10 is mapped to vn-segment 10010.
- VRF RED is the VRF associated with this VLAN.
- 20010 is the L3 VNI for VRF RED.
- VLAN 100 is mapped to L3 VNI 20010.

- Configure the Layer 2 VLAN.

```
vlan 10
  name RED
  vn-segment 10010
```

- Configure the VLAN for L3 VNI .

```
vlan 100
 name RED_L3_VNI_VLAN
 vn-segment 20010
```

- Define the anycast gateway MAC.

```
fabric forwarding anycast-gateway-mac 0000.2222.3333
```

- Define the NVE interface.

```
interface nve1
  no shutdown
  source-interface loopback1
  host-reachability protocol bgp
  member vni 10000 associate-vrf
  mcast-group 224.1.1.1
  member vni 10001 associate-vrf
  mcast-group 224.1.1.1
  member vni20000
  suppress-arp
  mcast-group 225.1.1.1
  member vni 20001
  suppress-arp
  mcast-group 225.1.1.1


evpn
  vni 10010 l2


rd auto
    route-target import auto
    route-target export auto
```

- Add configuration the to SVI definition on VLAN 10 and on L3 VNI VLAN 100.

```
interface Vlan10
  description RED
  no shutdown
  vrf member RED
  no ip redirects
  ip address 10.1.1.1/24
  ipv6 address 2001::1/64
  fabric forwarding mode anycast-gateway
```

- Configure SVI definition for VLAN 100.

```
interface Vlan100
 description RED_L3_VNI_VLAN
 no shutdown
 vrf member RED
 ip forward
 ipv6 address use-link-local-only
```

Note The IPv6 address use-link-local-only serves the same purpose as IP FORWARD for IPv4. It enables the switch to perform an IP based lookup even when the interface VLAN has no IP address defined under it.

- Add configuration to the VRF definition.

```
vrf context RED
  vni 20010


  rd auto
  address-family ipv4 unicast
```

```
    route-target both auto
    route-target both auto evpn
  address-family ipv6 unicast
    route-target both auto
    route-target both auto evpn


evpn
  vni 10010 l2


rd auto
    route-target import auto
    route-target export auto
```

- Add configuration to the VRF definition under BGP.

```
router bgp 65000
  vrf RED
    address-family ipv4 unicast
      advertise l2vpn evpn
   address-family ipv6 unicast
      advertise l2vpn evpn
```

Note If VTEPs are configured to operate as VPC peers, the following configuration is a best practice that should be included under the VPC domain on both switches.

```
vpc domain 1
   ipv6 nd synchronize
```

Show Command Examples

The following are examples of verifying IPv6 advertisement over VXLAN EVPN:

- Display ND information for the connected server.

```
9396-B_VTEP# show ipv6 neighbor vrf RED

Flags: # - Adjacencies Throttled for Glean
       G - Adjacencies of vPC peer with G/W bit
       R - Adjacencies learnt remotely

IPv6 Adjacency Table for VRF RED
Total number of entries: 2
Address         Age       MAC Address     Pref Source     Interface
2001::64        00:00:26  7c69.f614.2bc1  50   icmpv6     Vlan10
fe80::7e69:f6ff:fe14:2bc1
                00:01:13  7c69.f614.2bc1  50   icmpv6     Vlan10
```

- Check the L2ROUTE and ensure the MAC-IP was learned.

```
9396-B_VTEP# show l2route evpn mac-ip evi 10 host-ip 2001::64
Mac Address    Prod Host IP                                 Next Hop (s)
```

```
-------------- ---- -------------------------------------- ---------------
7c69.f614.2bc1 HMM  2001::64                               N/A
```

Note MAC-IP table is populated only when the end server sends a neighbor solicitation message (ARP in case of IPv4).

- Verify the route is present locally in the BGP table.

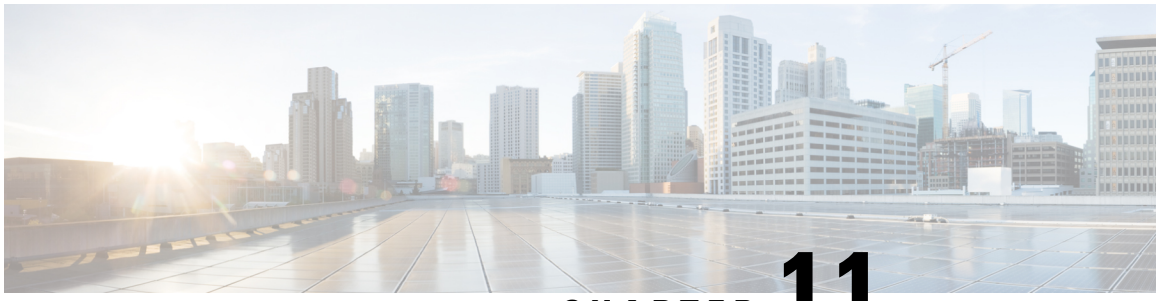
```
9396-B_VTEP# show bgp l2vpn evpn 2001::64
BGP routing table information for VRF default, address family L2VPN EVPN
Route Distinguisher: 198.19.0.15:34180    (L2VNI 10010)
BGP routing table entry for [2]:[0]:[0]:[48]:[7c69.f614.2bc1]:[128]:[2001::64]/368,
version 678
Paths: (1 available, best #1)
Flags: (0x00010a) on xmit-list, is not in l2rib/evpn

  Advertised path-id 1
  Path type: local, path is valid, is best path, no labeled nexthop
  AS-Path: NONE, path locally originated
    198.19.0.15 (metric 0) from 0.0.0.0 (198.19.0.15)
      Origin IGP, MED not set, localpref 100, weight 32768
      Received label 10010 20010
      Extcommunity:  RT:64567:10010 RT:64567:20010

    Path-id 1 advertised to peers:
    198.19.0.3
    198.19.0.4
```

- Verify the route is present in the remote VTEP 9396-A-VTEP BGP table.

```
9396-A-VTEP# show bgp l2vpn evpn 2001::64
BGP routing table information for VRF default, address family L2VPN EVPN
Route Distinguisher: 198.19.0.14:34180    (L2VNI 10010)
BGP routing table entry for [2]:[0]:[0]:[48]:[7c69.f614.2bc1]:[128]:[2001::64]/368,
version 305
Paths: (1 available, best #1)
Flags: (0x00021a) on xmit-list, is in l2rib/evpn, is not in HW,

  Advertised path-id 1
  Path type: internal, path is valid, is best path, no labeled nexthop
             Imported from
198.19.0.15:34180:[2]:[0]:[0]:[48]:[7c69.f614.2bc1]:[128]:[2001::64]/240
  AS-Path: NONE, path sourced internal to AS
    198.19.0.15 (metric 81) from 198.19.0.3 (198.19.0.3)
      Origin IGP, MED not set, localpref 100, weight 0
      Received label 10010 20010
      Extcommunity:  RT:64567:10010 RT:64567:20010 ENCAP:8 Router MAC:5087.89a1.a52f
      Originator: 198.19.0.15 Cluster list: 198.19.0.3
```

- Check the L2ROUTE and ensure that the MAC-IP was learned on the remote VTEP - 9396-A-VTEP.

```
rswV1leaf14# show l2route evpn mac-ip evi 1413 host-ip 2001::64
Mac Address    Prod Host IP                                 Next Hop (s)
-------------- ---- --------------------------------------- --------------
7c69.f614.2bc1 BGP  2001::64                               198.19.0.15
```



CHAPTER 11

Configuring Virtual Port Channels

This chapter contains the following sections:

- Information About vPCs, on page 223
- Guidelines and Limitations for vPCs, on page 231
- Verifying the vPC Configuration, on page 232
- vPC Default Settings, on page 238
- Configuring vPCs, on page 239
- Configuring Layer 3 over vPC, on page 253

Information About vPCs

vPC Overview

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

You configure the EtherChannels by using one of the following:

- No protocol
- Link Aggregation Control Protocol (LACP)

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

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

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

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

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

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

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

A vPC provides the following benefits:

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

Terminology

vPC Terminology

The terminology used in vPCs is as follows:

- vPC—combined EtherChannel between the vPC peer devices and the downstream device.
- vPC peer device—One of a pair of devices that are connected with the special EtherChannel known as the vPC peer link.
- vPC peer link—link used to synchronize states between the vPC peer devices.
- vPC member port—Interfaces that belong to the vPCs.
- vPC domain—domain that includes both vPC peer devices, the vPC peer-keepalive link, and all of the port channels in the vPC connected to the downstream devices. It is also associated to the configuration mode that you must use to assign vPC global parameters. The vPC domain ID must be the same on both switches.
- vPC peer-keepalive link—The peer-keepalive link monitors the vitality of a vPC peer Cisco Nexus device. The peer-keepalive link sends configurable, periodic keepalive messages between vPC peer devices.

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

vPC Domain

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

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

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

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

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

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

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

Peer-Keepalive Link and Messages

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

If one of the vPC peer switches fails, the vPC peer switch on the other side of the vPC peer link senses the failure when it does not receive any peer-keepalive messages. The default interval time for the vPC peer-keepalive message is 1 second. You can configure the interval between 400 milliseconds and 10 seconds. You can also configure a timeout value with a range of 3 to 20 seconds; the default timeout value is 5 seconds. The peer-keepalive status is checked only when the peer-link goes down.

The vPC peer-keepalive can be carried either in the management or default VRF on the Cisco Nexus device. When you configure the switches to use the management VRF, the source and destination for the keepalive messages are the mgmt 0 interface IP addresses. When you configure the switches to use the default VRF, an SVI must be created to act as the source and destination addresses for the vPC peer-keepalive messages. 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.

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

Compatibility Parameters for vPC Peer Links

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

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

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

New Type 2 Consistency Check on the vPC Port-Channels

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

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

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

Name                                Type            Local Value                 Peer Value
-------------                        ----            -----------------------
------------------------
Shut Lan                            1               No                          No
STP Port Type                       1               Default                      Default
STP Port Guard                      1               None                         None
STP MST Simulate PVST               1               Default                      Default
nve configuration                   1               nve                         nve
lag-id                              1               [(fa0,                       [(fa0,
                                                    0-23-4-ee-be-64, 8458,
0-23-4-ee-be-64, 8458,
                                                    0, 0), (8000,               0, 0),
(8000,
                                                    f4-4e-5-84-5e-3c, 457,
f4-4e-5-84-5e-3c, 457,
                                                    0, 0)]                      0, 0)]
mode                                1               active                       active
Speed                               1               10 Gb/s                     10 Gb/s
Duplex                              1               full                         full
Port Mode                           1               trunk                        trunk
Native Vlan                         1               1                           1
MTU                                 1               1500                         1500
Admin port mode                     1
Switchport MAC Learn                2               Enable                      Disable>
Newly added consistency parameter
vPC card type                       1               Empty                       Empty
```

```
Allowed VLANs                    -          311-400                  311-400
Local suspended VLANs            -          -
```

Configuration Parameters That Must Be Identical

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

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

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

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

- Port-channel mode: on, off, or active
- Link speed per channel
- Duplex mode per channel
- Trunk mode per channel:
  - Native VLAN
  - VLANs allowed on trunk
  - Tagging of native VLAN traffic
- Spanning Tree Protocol (STP) mode
- STP region configuration for Multiple Spanning Tree (MST)
- Enable or disable state per VLAN
- STP global settings:
  - Bridge Assurance setting
  - Port type setting—We recommend that you set all vPC interfaces as normal ports
  - Loop Guard settings
- STP interface settings:
  - Port type setting
  - Loop Guard
  - Root Guard

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

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

Configuration Parameters That Should Be Identical

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

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

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

Per-VLAN Consistency Check

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

vPC Auto-Recovery

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

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

The vPC auto-recovery feature is disabled by default.

vPC Peer Links

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

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

vPC Peer Link Overview

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

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

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

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

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

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

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

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

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

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

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

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

vPC Number

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

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

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

vPC Interactions with Other Features

vPC and LACP

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

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

The vPC peer link supports 16 EtherChannel interfaces.

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

vPC Peer Links and STP

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

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

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

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

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

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

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

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

CFSoE

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

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

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

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

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

Guidelines and Limitations for vPCs

vPCs have the following configuration guidelines and limitations:

- vPC is not supported between different types of Cisco Nexus 3000 Series switches.

-
- VPC peers should have same reserved VLANs for VXLAN. Different reserved VLANs on the peers may lead to undesired behavior with VXLAN.
- Starting with Release 7.0(3)I2(1), the output of the **sh vpc brief** CLI command displays two additional fields, Delay-restore status and Delay-restore SVI status.
- vPC is not qualified with IPv6.
- You must enable the vPC feature before you can configure vPC peer-link and vPC interfaces.
- You must configure the peer-keepalive link before the system can form the vPC peer link.
- The vPC peer-link needs to be formed using a minimum of two 10-Gigabit Ethernet interfaces.
- We recommend that you configure the same vPC domain ID on both peers and the domain ID should be unique in the network. For example, if there are two different vPCs (one in access and one in aggregation) then each vPC should have a unique domain ID.
- Only port channels can be in vPCs. A vPC can be configured on a normal port channel (switch-to-switch vPC topology) and on a port channel host interface (host interface vPC topology).
- You must configure both vPC peer switches; the configuration is not automatically synchronized between the vPC peer devices.
- Check that the necessary configuration parameters are compatible on both sides of the vPC peer link.
- You might experience minimal traffic disruption while configuring vPCs.
- You should configure all port channels in the vPC using LACP with the interfaces in active mode.
- You might experience traffic disruption when the first member of a vPC is brought up.
- OSPF over vPC and BFD with OSPF are supported on Cisco Nexus 3000 and 3100 Series switches.

  SVI limitation: When a BFD session is over SVI using virtual port-channel(vPC) peer-link, the BFD echo function is not supported. You must disable the BFD echo function for all sessions over SVI between vPC peer nodes using **no bfd echo** at the SVI configuration level.
- When a Layer 3 link is used for peer-keepalive instead of the mgmt interface, and the CPU queues are congested with control plane traffic, vPC peer-keepalive packets could be dropped. The CPU traffic includes routing protocol, ARP, Glean, and IPMC miss packets. When the peer-keepalive interface is a Layer 3 link instead of a mgmt interface, the vPC peer-keepalive packets are sent to the CPU on a low-priority queue.

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

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

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

Verifying the vPC Configuration

Use the following commands to display vPC configuration information:

Command	Purpose
switch# **show feature**	Displays whether vPC is enabled or not.
switch# **show port-channel capacity**	Displays how many EtherChannels are configured and how many are still available on the switch.
switch# **show running-config vpc**	Displays running configuration information for vPCs.
switch# **show vpc brief**	Displays brief information on the vPCs.
switch# **show vpc consistency-parameters**	Displays the status of those parameters that must be consistent across all vPC interfaces.
switch# **show vpc peer-keepalive**	Displays information on the peer-keepalive messages.
switch# **show vpc role**	Displays the peer status, the role of the local switch, the vPC system MAC address and system priority, and the MAC address and priority for the local vPC switch.
switch# **show vpc statistics**	Displays statistics on the vPCs. **Note** This command displays the vPC statistics only for the vPC peer device that you are working on.

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

Viewing the Graceful Type-1 Check Status

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

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

vPC domain id                     : 10
Peer status                       : peer adjacency formed ok
vPC keep-alive status             : peer is alive
Configuration consistency status: success
Per-vlan consistency status     : success
Type-2 consistency status       : success
vPC role                          : secondary
Number of vPCs configured       : 34
Peer Gateway                      : Disabled
Dual-active excluded VLANs      : -
Graceful Consistency Check      : Enabled
Auto-recovery status              : Disabled
Delay-restore status              : Timer is off.(timeout = 30s)
Delay-restore SVI status          : Timer is off.(timeout = 10s)

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

Viewing a Global Type-1 Inconsistency

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

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

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

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

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

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

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

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

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

vPC Peer-link status
---------------------------------------------------------------------
id   Port   Status Active vlans
--   ----   ------ --------------------------------------------------
```

```
1    Po1    up     1-10

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

Viewing an Interface-Specific Type-1 Inconsistency

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

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

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

vPC domain id                   : 10
Peer status                     : peer adjacency formed ok
vPC keep-alive status           : peer is alive
Configuration consistency status: success
Per-vlan consistency status     : success
Type-2 consistency status       : success
vPC role                        : secondary
Number of vPCs configured       : 2
Peer Gateway                    : Disabled
Dual-active excluded VLANs      : -
Graceful Consistency Check      : Enabled
Auto-recovery status              : Disabled
Delay-restore status              : Timer is off.(timeout = 30s)
Delay-restore SVI status          : Timer is off.(timeout = 10s)

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

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

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

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

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

```
Configuration consistency status: success
Per-vlan consistency status     : success
Type-2 consistency status       : success
vPC role                        : primary
Number of vPCs configured       : 2
Peer Gateway                    : Disabled
Dual-active excluded VLANs      : -
Graceful Consistency Check      : Enabled
Auto-recovery status              : Disabled
Delay-restore status              : Timer is off.(timeout = 30s)
Delay-restore SVI status          : Timer is off.(timeout = 10s)

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

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

Viewing a Per-VLAN Consistency Status

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

Example

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

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

vPC domain id                   : 10
Peer status                     : peer adjacency formed ok
vPC keep-alive status           : peer is alive
Configuration consistency status: success
Per-vlan consistency status     : success
Type-2 consistency status       : success
vPC role                        : secondary
Number of vPCs configured       : 2
Peer Gateway                    : Disabled
Dual-active excluded VLANs      : -
Graceful Consistency Check      : Enabled
Auto-recovery status              : Disabled
Delay-restore status              : Timer is off.(timeout = 30s)
Delay-restore SVI status          : Timer is off.(timeout = 10s)

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

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

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

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

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

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

vPC domain id                   : 10
Peer status                     : peer adjacency formed ok
vPC keep-alive status           : peer is alive
Configuration consistency status: success
Per-vlan consistency status     : failed
Type-2 consistency status       : success
vPC role                        : secondary
Number of vPCs configured       : 2
Peer Gateway                    : Disabled
Dual-active excluded VLANs      : -
Graceful Consistency Check      : Enabled
Auto-recovery status              : Disabled
Delay-restore status              : Timer is off.(timeout = 30s)
Delay-restore SVI status          : Timer is off.(timeout = 10s)

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

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

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

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

vPC domain id                   : 10
Peer status                     : peer adjacency formed ok
vPC keep-alive status           : peer is alive
Configuration consistency status: success
Per-vlan consistency status     : failed
Type-2 consistency status       : success
vPC role                        : primary
Number of vPCs configured       : 2
Peer Gateway                    : Disabled
```

```
Dual-active excluded VLANs      : -
Graceful Consistency Check      : Enabled
Auto-recovery status              : Disabled
Delay-restore status              : Timer is off.(timeout = 30s)
Delay-restore SVI status          : Timer is off.(timeout = 10s)

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

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

This example shows the inconsistency as STP Disabled:

```
switch(config)# show vpc consistency-parameters vlans

Name                        Type  Reason Code             Pass Vlans

-------------               ----  --------------------- -----------------------
STP Mode                    1     success               0-4095
STP Disabled                1     vPC type-1            0-4,6-4095
                                  configuration
                                  incompatible - STP is
                                  enabled or disabled on
                                   some or all vlans
STP MST Region Name         1     success               0-4095
STP MST Region Revision     1     success               0-4095
STP MST Region Instance to  1     success               0-4095
 VLAN Mapping
STP Loopguard               1     success               0-4095
STP Bridge Assurance        1     success               0-4095
STP Port Type, Edge         1     success               0-4095
BPDUFilter, Edge BPDUGuard
STP MST Simulate PVST       1     success               0-4095
Pass Vlans                  -                           0-4,6-4095
```

vPC Default Settings

The following table lists the default settings for vPC parameters.

Table 10: 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

Parameters	Default
vPC peer-keepalive UDP port	3200

Configuring vPCs

Enabling vPCs

You must enable the vPC feature before you can configure and use vPCs.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **feature vpc**	Enables vPCs on the switch.
Step 3	(Optional) switch# **show feature**	Displays which features are enabled on the switch.
Step 4	(Optional) switch# **copy running-config startup-config**	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
```

Disabling vPCs

You can disable the vPC feature.

Note When you disable the vPC feature, the Cisco Nexus device clears all the vPC configurations.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **no feature vpc**	Disables vPCs on the switch.

	Command or Action	Purpose
Step 3	(Optional) switch# **show feature**	Displays which features are enabled on the switch.
Step 4	(Optional) switch# **copy running-config startup-config**	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
```

Creating a vPC Domain

You must create identical vPC domain IDs on both the vPC peer devices. This domain ID is used to automatically form the vPC system MAC address.

Before you begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **vpc domain** *domain-id*	Creates a vPC domain on the switch, and enters the vpc-domain configuration mode. There is no default *domain-id* ; the range is from 1 to 1000. **Note** You can also use the **vpc domain** command to enter the vpc-domain configuration mode for an existing vPC domain.
Step 3	(Optional) switch# **show vpc brief**	Displays brief information about each vPC domain.
Step 4	(Optional) switch# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to create a vPC domain:

```
switch# configure terminal
switch(config)# vpc domain 5
```

Configuring a vPC Keepalive Link and Messages

You can configure the destination IP for the peer-keepalive link that carries the keepalive messages. Optionally, you can configure other parameters for the keepalive messages.

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

Ensure that both the source and destination IP addresses used for the peer-keepalive message are unique in your network and these IP addresses are reachable from the Virtual Routing and Forwarding (VRF) instance associated with the vPC peer-keepalive link.

Note We recommend that you configure a separate VRF instance and put a Layer 3 port from each vPC peer switch into that VRF instance for the vPC peer-keepalive link. Do not use the peer link itself to send vPC peer-keepalive messages.

Before you begin

Ensure that you have enabled the vPC feature.

You must configure the vPC peer-keepalive link before the system can form the vPC peer link.

You must configure both switches on either side of the vPC peer link.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **vpc domain** *domain-id*	Creates a vPC domain on the switch if it does not already exist, and enters the vpc-domain configuration mode.
Step 3	switch(config-vpc-domain)# **peer-keepalive destination** *ipaddress* [**hold-timeout** *secs* | **interval** *msecs* {**timeout** *secs*} | **precedence** {*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**}]	Configures the IPv4 address 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. The management ports and VRF are the defaults.

	Command or Action	Purpose
Step 4	(Optional) switch(config-vpc-domain)# **vpc peer-keepalive destination** *ipaddress* **source** *ipaddress*	Configures a separate VRF instance and puts a Layer 3 port from each vPC peer device into that VRF for the vPC peer-keepalive link.
Step 5	(Optional) switch# **show vpc peer-keepalive**	Displays information about the configuration for the keepalive messages.
Step 6	(Optional) switch# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to configure the destination IP address for the vPC-peer-keepalive link:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# peer-keepalive destination 10.10.10.42
```

This example shows how to set up the peer keepalive link connection between the primary and secondary vPC device:

```
switch(config)# vpc domain 100
switch(config-vpc-domain)# peer-keepalive destination 192.168.2.2 source 192.168.2.1
Note:--------:: Management VRF will be used as the default VRF ::--------
switch(config-vpc-domain)#
```

This example shows how to create a separate VRF named vpc_keepalive for the vPC keepalive link and how to verify the new VRF:

```
vrf context vpc_keepalive
interface Ethernet1/31
  switchport access vlan 123
interface Vlan123
  vrf member vpc_keepalive
  ip address 123.1.1.2/30
  no shutdown
vpc domain 1
  peer-keepalive destination 123.1.1.1 source 123.1.1.2 vrf
vpc_keepalive

L3-NEXUS-2# show vpc peer-keepalive

vPC keep-alive status           : peer is alive
--Peer is alive for             : (154477) seconds, (908) msec
--Send status                   : Success
--Last send at                  : 2011.01.14 19:02:50 100 ms
--Sent on interface             : Vlan123
--Receive status                : Success
--Last receive at               : 2011.01.14 19:02:50 103 ms
--Received on interface         : Vlan123
--Last update from peer         : (0) seconds, (524) msec

vPC Keep-alive parameters
--Destination                   : 123.1.1.1
--Keepalive interval            : 1000 msec
```

```
--Keepalive timeout             : 5 seconds
--Keepalive hold timeout        : 3 seconds
--Keepalive vrf                 : vpc_keepalive
--Keepalive udp port            : 3200
--Keepalive tos                 : 192

The services provided by the switch , such as ping, ssh, telnet,
radius, are VRF aware. The VRF name need to be configured or
specified in order for the correct routing table to be used.
L3-NEXUS-2# ping 123.1.1.1 vrf vpc_keepalive
PING 123.1.1.1 (123.1.1.1): 56 data bytes
64 bytes from 123.1.1.1: icmp_seq=0 ttl=254 time=3.234 ms
64 bytes from 123.1.1.1: icmp_seq=1 ttl=254 time=4.931 ms
64 bytes from 123.1.1.1: icmp_seq=2 ttl=254 time=4.965 ms
64 bytes from 123.1.1.1: icmp_seq=3 ttl=254 time=4.971 ms
64 bytes from 123.1.1.1: icmp_seq=4 ttl=254 time=4.915 ms

--- 123.1.1.1 ping statistics ---
5 packets transmitted, 5 packets received, 0.00% packet loss
round-trip min/avg/max = 3.234/4.603/4.971 ms
```

Creating a vPC Peer Link

You can create a vPC peer link by designating the EtherChannel that you want on each switch as the peer link for the specified vPC domain. We recommend that you configure the EtherChannels that you are designating as the vPC peer link in trunk mode and that you use two ports on separate modules on each vPC peer switch for redundancy.

Before you begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface port-channel** *channel-number*	Selects the EtherChannel that you want to use as the vPC peer link for this switch, and enters the interface configuration mode.
Step 3	switch(config-if)# **vpc peer-link**	Configures the selected EtherChannel as the vPC peer link, and enters the vpc-domain configuration mode.
Step 4	(Optional) switch# **show vpc brief**	Displays information about each vPC, including information about the vPC peer link.
Step 5	(Optional) switch# **copy running-config startup-config**	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)# vpc peer-link
```

Checking the Configuration Compatibility

After you have configured the vPC peer link on both vPC peer switches, check that the configurations are consistent on all vPC interfaces.

The following QoS parameters support Type 2 consistency checks

- Network QoS—MTU and Pause
- Input Queuing —Bandwidth and Absolute Priority
- Output Queuing—Bandwidth and Absolute Priority

In the case of a Type 2 mismatch, the vPC is not suspended. Type 1 mismatches suspend the vPC.

Procedure

	Command or Action	Purpose
Step 1	switch# **show vpc consistency-parameters** {**global**|**interface port-channel***channel-number*}	Displays the status of those parameters that must be consistent across all vPC interfaces.

Example

This example shows how to check that the required configurations are compatible across all the vPC interfaces:

```
switch# show vpc consistency-parameters global
Legend:
        Type 1 : vPC will be suspended in case of mismatch
Name                        Type  Local Value            Peer Value

------------- ----  ---------------------- -----------------------
QoS                         2     ([], [], [], [], [],   ([], [], [], [], [],
                                   [])                    [])

Network QoS (MTU)           2     (1538, 0, 0, 0, 0, 0)  (1538, 0, 0, 0, 0, 0)
Network Qos (Pause)         2     (F, F, F, F, F, F)     (1538, 0, 0, 0, 0, 0)
Input Queuing (Bandwidth)   2     (100, 0, 0, 0, 0, 0)   (100, 0, 0, 0, 0, 0)
Input Queuing (Absolute     2     (F, F, F, F, F, F)     (100, 0, 0, 0, 0, 0)
Priority)
Output Queuing (Bandwidth)  2     (100, 0, 0, 0, 0, 0)   (100, 0, 0, 0, 0, 0)
Output Queuing (Absolute    2     (F, F, F, F, F, F)     (100, 0, 0, 0, 0, 0)
Priority)
STP Mode                    1     Rapid-PVST             Rapid-PVST
STP Disabled                1     None                   None
STP MST Region Name         1     ""                     ""
```

```
STP MST Region Revision     1     0                    0
STP MST Region Instance to  1
  VLAN Mapping

STP Loopguard               1     Disabled             Disabled
STP Bridge Assurance        1     Enabled              Enabled
STP Port Type, Edge         1     Normal, Disabled,    Normal, Disabled,
BPDUFilter, Edge BPDUGuard        Disabled             Disabled
STP MST Simulate PVST       1     Enabled              Enabled
Allowed VLANs               -     1,624                1
Local suspended VLANs       -     624                  -
switch#
```

Enabling vPC Auto-Recovery

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **vpc domain** *domain-id*	Enters vpc-domain configuration mode for an existing vPC domain.
Step 3	switch(config-vpc-domain)# **auto-recovery reload-delay** *delay*	Enables the auto-recovery feature and sets the reload delay period. The default is disabled.

Example

This example shows how to enable the auto-recovery feature in vPC domain 10 and set the delay period for 240 seconds:

```
switch(config)# vpc domain 10
switch(config-vpc-domain)# auto-recovery reload-delay 240
Warning:
 Enables restoring of vPCs in a peer-detached state after reload, will wait for 240 seconds
 (by default) to determine if peer is un-reachable
```

This example shows how to view the status of the auto-recovery feature in vPC domain 10:

```
switch(config-vpc-domain)# show running-config vpc
!Command: show running-config vpc
!Time: Tue Dec  7 02:38:44 2010


version 5.0(3)U2(1)
feature vpc
vpc domain 10
  peer-keepalive destination 10.193.51.170
  auto-recovery
```

Configuring the Restore Time Delay

You can configure a restore timer that delays the vPC from coming back up until after the peer adjacency forms and the VLAN interfaces are back up. This feature avoids packet drops if the routing tables fail to converge before the vPC is once again passing traffic.

Before you begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link with the following procedures.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **vpc domain** *domain-id*	Creates a vPC domain on the switch if it does not already exist, and enters vpc-domain configuration mode.
Step 3	switch(config-vpc-domain)# **delay restore** *time*	Configures the time delay before the vPC is restored. The restore time is the number of seconds to delay bringing up the restored vPC peer device. The range is from 1 to 3600. The default is 30 seconds.
Step 4	(Optional) switch# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to configure the delay reload time for a vPC link:

```
switch(config)# vpc domain 1
switch(config-vpc-domain)# delay restore 10
switch(config-vpc-domain)#
```

Configuring Delay Restore on an Orphan Port

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**switch(config) # vpc domain <domain>**	Configure the VPC domain number.
Step 3	**switch(config) # peer-switch**	Define the peer switch.

	Command or Action	Purpose
Step 4	**switch(config) # show vpc peer-keepalive**	Displays information about the peer keepalive messages
Step 5	**switch(config) # delay restore { time }**	Number of seconds to delay bringing up the restored vPC peer device. The range is from 1 to 3600.
Step 6	**switch(config) # peer-gateway**	To enable Layer 3 forwarding for packets destined to the gateway MAC address of the virtual Port Channel (vPC), use the peer-gateway command. To disable Layer 3 forwarding packets, use the no form of this command. .
Step 7	**switch(config) # delay restore orphan-port**	Number of seconds to delay bringing up the restored device's orphan port

Configuring the Suspension of Orphan Ports

Note You can configure vPC orphan port suspension only on physical ports, not on port channel member ports.

Before you begin

Ensure that you have enabled the vPC feature.

Ensure that you are in the correct VDC (or use the switchto vdc command).

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(config)# show vpc orphan-ports`
	`switch(config-vpc-domain)#`
		(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.

	Command or Action	Purpose
Step 4	**vpc orphan-ports 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 case of vPC failure.
Step 5	**exit** **Example:** `switch(config-if)# exit`	Exits the 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 case of vPC failure:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# vpc orphan-ports suspend
switch(config-if)#
```

Excluding VLAN Interfaces from Shutting Down a vPC Peer Link Fails

When a vPC peer-link is lost, the vPC secondary switch suspends its vPC member ports and its switch virtual interface (SVI) interfaces. All Layer 3 forwarding is disabled for all VLANs on the vPC secondary switch. You can exclude specific SVI interfaces so that they are not suspended.

Before you begin

Ensure that the VLAN interfaces have been configured.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **vpc domain** *domain-id*	Creates a vPC domain on the switch if it does not already exist, and enters vpc-domain configuration mode.

	Command or Action	Purpose
Step 3	switch(config-vpc-domain))# **dual-active exclude interface-vlan** *range*	Specifies the VLAN interfaces that should remain up when a vPC peer-link is lost. range—Range of VLAN interfaces that you want to exclude from shutting down. The range is from 1 to 4094.

Example

This example shows how to keep the interfaces on VLAN 10 up on the vPC peer switch if a peer link fails:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# dual-active exclude interface-vlan 10
switch(config-vpc-domain)#
```

Configuring the VRF Name

The switch services, such as ping, ssh, telnet, radius, are VRF aware. You must configure the VRF name in order for the correct routing table to be used.

You can specify the VRF name.

Procedure

	Command or Action	Purpose
Step 1	switch# **ping** *ipaddress* **vrf** *vrf-name*	Specifies the virtual routing and forwarding (VRF) name to use. The VRF name is case sensitive and can be a maximum of 32 characters..

Example

This example shows how to specifiy the VRF named vpc_keepalive:

```
switch# ping 123.1.1.1 vrf vpc_keepalive
PING 123.1.1.1 (123.1.1.1): 56 data bytes
64 bytes from 123.1.1.1: icmp_seq=0 ttl=254 time=3.234 ms
64 bytes from 123.1.1.1: icmp_seq=1 ttl=254 time=4.931 ms
64 bytes from 123.1.1.1: icmp_seq=2 ttl=254 time=4.965 ms
64 bytes from 123.1.1.1: icmp_seq=3 ttl=254 time=4.971 ms
64 bytes from 123.1.1.1: icmp_seq=4 ttl=254 time=4.915 ms

--- 123.1.1.1 ping statistics ---
5 packets transmitted, 5 packets received, 0.00% packet loss
round-trip min/avg/max = 3.234/4.603/4.971 ms
```

Moving Other Port Channels into a vPC

Before you begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link with the following procedure.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface port-channel** *channel-number*	Selects the port channel that you want to put into the vPC to connect to the downstream switch, and enters interface configuration mode. **Note** A vPC can be configured on a normal port channel (physical vPC topology) and on a port channel host interface (host interface vPC topology)
Step 3	switch(config-if)# **vpc** *number*	Configures the selected port channel into the vPC to connect to the downstream switch. The range is from 1 to 4096. The vPC *number* that you assign to the port channel that connects to the downstream switch from the vPC peer switch must be identical on both vPC peer switches.
Step 4	(Optional) switch# **show vpc brief**	Displays information about each vPC.
Step 5	(Optional) switch# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to configure a port channel that will connect to the downstream device:

```
switch# configure terminal
switch(config)# interface port-channel 20
switch(config-if)# vpc 5
```

Manually Configuring a vPC Domain MAC Address

Note Configuring the system address is an optional configuration step.

Before you begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **vpc domain** *domain-id*	Selects an existing vPC domain on the switch, or creates a new vPC domain, and enters the vpc-domain configuration mode. There is no default *domain-id*; the range is from 1 to 1000.
Step 3	switch(config-vpc-domain)# **system-mac** *mac-address*	Enters the MAC address that you want for the specified vPC domain in the following format: aaaa.bbbb.cccc.
Step 4	(Optional) switch# **show vpc role**	Displays the vPC system MAC address.
Step 5	(Optional) switch# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to configure a vPC domain MAC address:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-if)# system-mac 23fb.4ab5.4c4e
```

Manually Configuring the System Priority

When you create a vPC domain, the system automatically creates a vPC system priority. However, you can also manually configure a system priority for the vPC domain.

Before you begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **vpc domain** *domain-id*	Selects an existing vPC domain on the switch, or creates a new vPC domain, and enters the vpc-domain configuration mode. There is no default *domain-id*; the range is from 1 to 1000.
Step 3	switch(config-vpc-domain)# **system-priority** *priority*	Enters the system priority that you want for the specified vPC domain. The range of values is from 1 to 65535. The default value is 32667.
Step 4	(Optional) switch# **show vpc brief**	Displays information about each vPC, including information about the vPC peer link.
Step 5	(Optional) switch# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to configure a vPC peer link:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-if)# system-priority 4000
```

Manually Configuring a vPC Peer Switch Role

By default, the Cisco NX-OS software elects a primary and secondary vPC peer switch after you configure the vPC domain and both sides of the vPC peer link. However, you may want to elect a specific vPC peer switch as the primary switch for the vPC. Then, you would manually configure the role value for the vPC peer switch that you want as the primary switch to be lower than the other vPC peer switch.

vPC does not support role preemption. If the primary vPC peer switch fails, the secondary vPC peer switch takes over to become operationally the vPC primary switch. However, the original operational roles are not restored when the formerly primary vPC comes up again.

Before you begin

Ensure that you have enabled the vPC feature.

You must configure both switches on either side of the vPC peer link.

Procedure

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

	Command or Action	Purpose
Step 2	switch(config)# **vpc domain** *domain-id*	Selects an existing vPC domain on the switch, or creates a new vPC domain, and enters the vpc-domain configuration mode. There is no default *domain-id*; the range is from 1 to 1000.
Step 3	switch(config-vpc-domain)# **role priority** *priority*	Enters the role priority that you want for the vPC system priority. The range of values is from 1 to 65535. The default value is 32667.
Step 4	(Optional) switch# **show vpc brief**	Displays information about each vPC, including information about the vPC peer link.
Step 5	(Optional) switch# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Example

This example shows how to configure a vPC peer link:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-if)# role priority 4000
```

Configuring Layer 3 over vPC

Before you begin

Ensure that the peer-gateway feature is enabled and it is configured on both the peers and both the peers run an image that supports Layer 3 over vPC. If you enter the **layer3 peer-router** command without enabling the peer-gateway feature, a syslog message is displayed recommending you to enable the peer-gateway feature.

Ensure that the vPC Peer-Link is up.

Procedure

	Command or Action	Purpose

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

Step 2	switch(config)# **vpc domain** *domain-id*
	Example:
	`switch(config)# vpc domain 5`
	`switch(config-vpc-domain)#`
		Creates a vPC domain if it does not already exist, and enters the vpc-domain configuration mode. There is no default; the range is from <1 to 1000>.

	Command or Action	Purpose
Step 3	switch(config-vpc-domain)#**layer3 peer-router**	Enables the Layer 3 device to form peering adjacency with both the peers. **Note** Configure this command in both the peers. If you configure this command only on one of the peers or you disable it on one peer, the operational state of layer 3 peer-router gets disabled. You get a notification when there is a change in the operational state.
Step 4	switch(config-vpc-domain)# **exit**	Exits the vpc-domain configuration mode.
Step 5	(Optional) switch#**show vpc brief**	Displays brief information about each vPC domain.
Step 6	(Optional) switch#**copy running-config startup-config**	Copies the running configuration to the startup configuration.

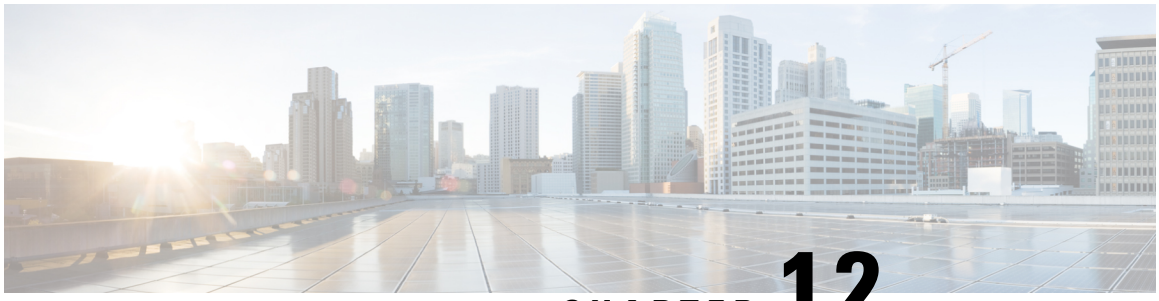
Example

The following example shows how to configure Layer 3 over vPC feature:

```
switch# configure terminal
switch(config)# vpc domain 5
switch(config-vpc-domain)# layer3 peer-router

switch(config-vpc-domain)# exit

switch(config)#
```

This example shows how to verify if the Layer 3 over vPC feature is configured. The **Operational Layer3 Peer** is enabled or disabled depending up on how the operational state of Layer 3 over vPC is configured.

```
switch# show vpc brief

vPC domain id : 5
Peer status : peer adjacency formed ok
vPC keep-alive status : peer is alive
Configuration consistency status : success
Per-vlan consistency status : failed
Type-2 consistency status : success
vPC role : secondary
Number of vPCs configured : 2
Peer Gateway : Enabled
Peer gateway excluded VLANs : -
Dual-active excluded VLANs : -
Graceful Consistency Check : Enabled
Auto-recovery status : Enabled (timeout = 240 seconds)
Operational Layer3 Peer : Enabled
```



CHAPTER 12

Configuring Q-in-Q VLAN Tunnels

This chapter contains the following sections:

- Information About Q-in-Q Tunnels, on page 255
- Information About Layer 2 Protocol Tunneling, on page 258
- Guidelines and Limitations for Q-in-Q Tunneling, on page 260
- Configuring Q-in-Q Tunnels and Layer 2 Protocol Tunneling, on page 261
- Verifying the Q-in-Q Configuration, on page 265
- Configuration Example for Q-in-Q and Layer 2 Protocol Tunneling, on page 265
- Feature History for Q-in-Q Tunnels and Layer 2 Protocol Tunneling, on page 266

Information About Q-in-Q Tunnels

A Q-in-Q VLAN tunnel enables a service provider to segregate the traffic of different customers in their infrastructure, while still giving the customer a full range of VLANs for their internal use by adding a second 802.1Q tag to an already tagged frame.

Business customers of service providers often have specific requirements for VLAN IDs and the number of VLANs to be supported. The VLAN ranges required by different customers in the same service-provider network might overlap, and traffic of customers through the infrastructure might be mixed. Assigning a unique range of VLAN IDs to each customer would restrict customer configurations and could easily exceed the VLAN limit of 4096 of the 802.1Q specification.

Note Q-in-Q is supported on port channels. To configure a port channel as an asymmetrical link, all ports in the port channel must have the same tunneling configuration.

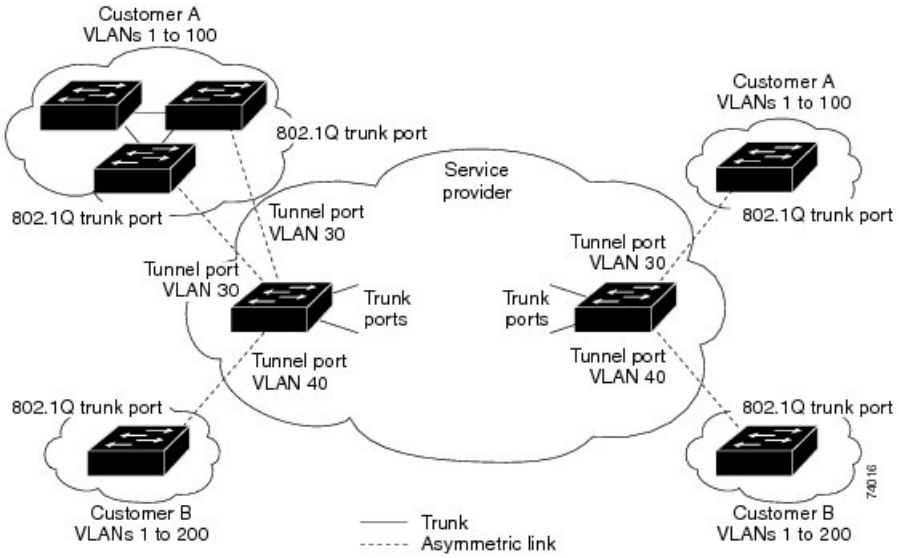
Using the 802.1Q tunneling feature, service providers can use a single VLAN to support customers who have multiple VLANs. Customer VLAN IDs are preserved and traffic from different customers is segregated within the service-provider infrastructure even when they appear to be on the same VLAN. The 802.1Q tunneling expands VLAN space by using a VLAN-in-VLAN hierarchy and tagging the tagged packets. A port configured to support 802.1Q tunneling is called a tunnel port. When you configure tunneling, you assign a tunnel port to a VLAN that is dedicated to tunneling. Each customer requires a separate VLAN, but that VLAN supports all of the customer's VLANs.

Customer traffic tagged in the normal way with appropriate VLAN IDs come from an 802.1Q trunk port on the customer device and into a tunnel port on the service-provider edge switch. The link between the customer

device and the edge switch is an asymmetric link because one end is configured as an 802.1Q trunk port and the other end is configured as a tunnel port. You assign the tunnel port interface to an access VLAN ID that is unique to each customer.

Note Selective Q-in-Q tunneling is not supported. All frames entering the tunnel port are subjected to Q-in-Q tagging.

Figure 29: 802.1Q-in-Q Tunnel Ports

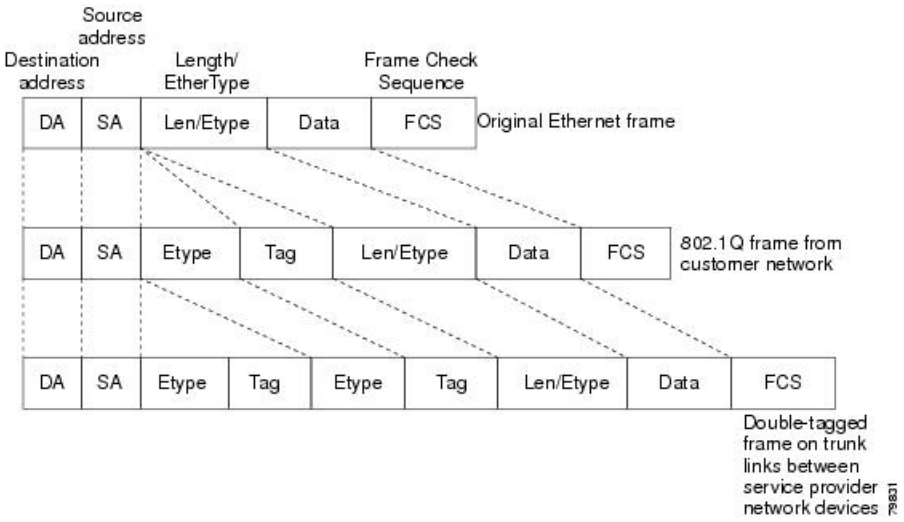


Packets that enter the tunnel port on the service-provider edge switch, which are already 802.1Q-tagged with the appropriate VLAN IDs, are encapsulated with another layer of an 802.1Q tag that contains a VLAN ID that is unique to the customer. The original 802.1Q tag from the customer is preserved in the encapsulated packet. Therefore, packets that enter the service-provider infrastructure are double-tagged.

The outer tag contains the customer's access VLAN ID (as assigned by the service provider), and the inner VLAN ID is the VLAN of the incoming traffic (as assigned by the customer). This double tagging is called tag stacking, Double-Q, or Q-in-Q.

The following figure shows the differences between the untagged, tagged and double-tagged ethernet frames.

Figure 30: Untagged, 802.1Q-Tagged, and Double-Tagged Ethernet Frames



By using this method, the VLAN ID space of the outer tag is independent of the VLAN ID space of the inner tag. A single outer VLAN ID can represent the entire VLAN ID space for an individual customer. This technique allows the customer's Layer 2 network to extend across the service provider network, potentially creating a virtual LAN infrastructure over multiple sites.

Note Hierarchical tagging, that is multi-level dot1q tagging Q-in-Q, is not supported.

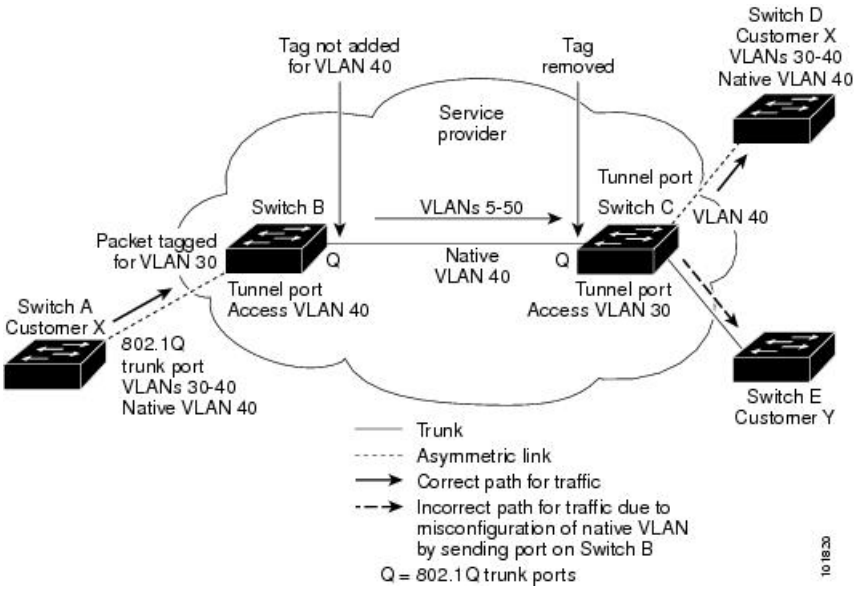
Native VLAN Hazard

When configuring 802.1Q tunneling on an edge switch, you must use 802.1Q trunk ports for sending out packets into the service-provider network. However, packets that go through the core of the service-provider network might be carried through 802.1Q trunks, ISL trunks, or nontrunking links. When 802.1Q trunks are used in these core switches, the native VLANs of the 802.1Q trunks must not match any native VLAN of the dot1q-tunnel port on the same switch because traffic on the native VLAN is not tagged on the 802.1Q transmitting trunk port.

VLAN 40 is configured as the native VLAN for the 802.1Q trunk port from Customer X at the ingress edge switch in the service-provider network (Switch B). Switch A of Customer X sends a tagged packet on VLAN 30 to the ingress tunnel port of Switch B in the service-provider network that belongs to access VLAN 40. Because the access VLAN of the tunnel port (VLAN 40) is the same as the native VLAN of the edge-switch trunk port (VLAN 40), the 802.1Q tag is not added to the tagged packets that are received from the tunnel port. The packet carries only the VLAN 30 tag through the service-provider network to the trunk port of the egress-edge switch (Switch C) and is misdirected through the egress switch tunnel port to Customer Y.

The following figure shows the native VLAN hazard.

Figure 31: Native VLAN Hazard



A couple of ways to solve the native VLAN problem, are as follows:

- Configure the edge switch so that all packets going out an 802.1Q trunk, including the native VLAN, are tagged by using the **vlan dot1q tag native** command. If the switch is configured to tag native VLAN packets on all 802.1Q trunks, the switch accepts untagged packets but sends only tagged packets.

Note The **vlan dot1q tag native** command is a global command that affects the tagging behavior on all trunk ports.

- Ensure that the native VLAN ID on the edge switch trunk port is not within the customer VLAN range. For example, if the trunk port carries traffic of VLANs 100 to 200, assign the native VLAN a number outside that range.

Information About Layer 2 Protocol Tunneling

Customers at different sites connected across a service-provider network need to run various Layer 2 protocols to scale their topology to include all remote sites, as well as the local sites. The Spanning Tree Protocol (STP) must run properly, and every VLAN should build a proper spanning tree that includes the local site and all remote sites across the service-provider infrastructure. Cisco Discovery Protocol (CDP) must be able to discover neighboring Cisco devices from local and remote sites, and the VLAN Trunking Protocol (VTP) must provide consistent VLAN configuration throughout all sites in the customer network.

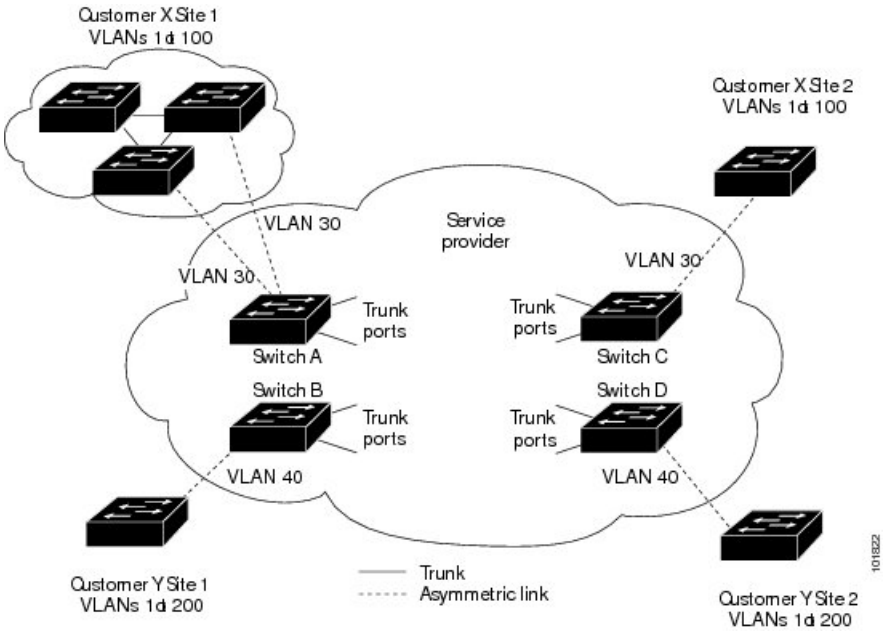
When protocol tunneling is enabled, edge switches on the inbound side of the service-provider infrastructure encapsulate Layer 2 protocol packets with a special MAC address and send them across the service-provider network. Core switches in the network do not process these packets, but forward them as normal packets. Bridge protocol data units (BPDUs) for CDP, STP, or VTP cross the service-provider infrastructure and are delivered to customer switches on the outbound side of the service-provider network. Identical packets are received by all customer ports on the same VLANs.

If protocol tunneling is not enabled on 802.1Q tunneling ports, remote switches at the receiving end of the service-provider network do not receive the BPDUs and cannot properly run STP, CDP, 802.1X, and VTP. When protocol tunneling is enabled, Layer 2 protocols within each customer's network are totally separate from those running within the service-provider network. Customer switches on different sites that send traffic through the service-provider network with 802.1Q tunneling achieve complete knowledge of the customer's VLAN.

Note Layer 2 protocol tunneling works by tunneling BPDUs in the software. A large number of BPDUs that comes into the supervisor module cause the CPU load to go up. The load It is controlled by Control Plane Policing CoPP configured for packets marked as BPDU.

For example, the following figure shows Customer X has four switches in the same VLAN that are connected through the service-provider network. If the network does not tunnel BPDUs, the switches on the far ends of the network cannot properly run the STP, CDP, 802.1X, and VTP protocols.

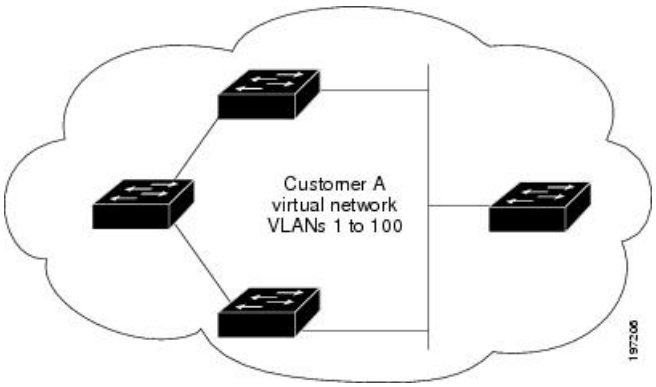
Figure 32: Layer 2 Protocol Tunneling



In the preceding example, STP for a VLAN on a switch in Customer X, Site 1 will build a spanning tree on the switches at that site without considering convergence parameters based on Customer X's switch in Site 2.

The following figure shows the resulting topology on the customer's network when BPDU tunneling is not enabled.

Figure 33: Virtual Network Topology Without BPDU Tunneling



Guidelines and Limitations for Q-in-Q Tunneling

Q-in-Q tunnels and Layer 2 tunneling have the following configuration guidelines and limitations:

- Cisco Nexus 3500 Series switches do not support Q-in-Q tunneling. However they forward Q-in-Q traffic.
- Cisco Nexus 3000 Series switches (except for the Nexus 3500 Series switches) do not support configuring Q-in-Q Tunneling on Cisco NX-OS Release 7.0(3)I7(2) and the previous releases.
- Switches in the service-provider network must be configured to handle the increase in MTU size due to Q-in-Q tagging.
- Selective Q-in-Q tunneling is not supported. All frames that enter the tunnel port will be subject to Q-in-Q tagging.
- MAC address learning for Q-in-Q tagged packets is based on the outer VLAN (Service Provider VLAN) tag. Packet forwarding issues may occur in deployments where a single MAC address is used across multiple inner (customer) VLANs.
- Layer 3 and higher parameters cannot be identified in tunnel traffic (for example, Layer 3 destination and source addresses). Tunneled traffic cannot be routed.
- You should use MAC address-based frame distribution.
- You cannot configure the 802.1Q tunneling feature on ports that are configured to support private VLANs. Private VLAN are not required in these deployments.
- CDP must be explicitly disabled, as needed, on the dot1Q tunnel port.
- You must disable IGMP snooping on the tunnel VLANs.
- You should run the **vlan dot1Q tag native** command to maintain the tagging on the native VLAN and drop untagged traffic to prevent native VLAN misconfigurations.
- You must manually configure the 802.1Q interfaces to be edge ports.
- Dot1x tunneling is not supported.

Configuring Q-in-Q Tunnels and Layer 2 Protocol Tunneling

Creating a 802.1Q Tunnel Port

You create the dot1q-tunnel port using the **switchport** mode command.

Note You must set the 802.1Q tunnel port to an edge port with the **spanning-tree port type edge** command. The VLAN membership of the port is changed when you enter the **switchport access vlan vlan-id** command.

You should disable IGMP snooping on the access VLAN allocated for the dot1q-tunnel port to allow multicast packets to traverse the Q-in-Q tunnel.

Before you begin

You must first configure the interface as a switchport.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface ethernet** *slot*/*port*	Specifies an interface to configure, and enters interface configuration mode.
Step 3	switch(config-if)# **switchport**	Sets the interface as a Layer 2 switching port.
Step 4	switch(config-if)# [**no**] **switchport mode dot1q-tunnel**	Creates an 802.1Q tunnel on the port. The port will go down and reinitialize (port flap) when the interface mode is changed. BPDU filtering is enabled and CDP is disabled on tunnel interfaces.
Step 5	switch(config-if)# **exit**	Exits interface configuration mode.
Step 6	(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 7	(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 create an 802.1Q tunnel port:

```
switch# configure terminal
switch(config)# interface ethernet 7/1
switch(config-if)# switchport
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# exit
switch(config)# exit
switch# show dot1q-tunnel
```

Enabling the Layer 2 Protocol Tunnel

You can enable protocol tunneling on the 802.1Q tunnel port.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface ethernet** *slot*/*port*	Specifies an interface to configure, and enters interface configuration mode.
Step 3	switch(config)# **switchport**	Sets the interface as a Layer 2 switching port.
Step 4	switch(config-if)# **switchport mode dot1q-tunnel**	Creates an 802.1Q tunnel on the port.
Step 5	switch(config-if)# [**no**] **l2protocol tunnel** [**cdp** | **stp** | **vtp**]	Enables Layer 2 protocol tunneling. Optionally, you can enable CDP, STP, or VTP tunneling.
Step 6	switch(config-if)# **exit**	Exits interface configuration mode.
Step 7	(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 enable protocol tunneling on an 802.1Q tunnel port:

```
switch# configure terminal
switch(config)# interface ethernet 7/1
switch(config-if)# switchport
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# l2protocol tunnel stp
switch(config-if)# exit
switch(config)# exit
```

Configuring Thresholds for Layer 2 Protocol Tunnel Ports

You can specify the port drop and shutdown value for a Layer 2 protocol tunneling port.

Procedure

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	switch(config)# **interface ethernet** *slot*/*port*	Specifies an interface to configure, and enters interface configuration mode.
Step 3	switch(config-if)# **switchport**	Sets the interface as a Layer 2 switching port.
Step 4	switch(config-if)# **switchport mode dot1q-tunnel**	Creates an 802.1Q tunnel on the port.
Step 5	switch(config-if)# [**no**] **l2protocol tunnel drop-threshold** [**cdp** | **stp** | **vtp**]	Specifies the maximum number of packets that can be processed on an interface before being dropped. Optionally, you can specify CDP, STP, or VTP. Valid values for the packets are from 1 to 4096. The **no** form of this command resets the threshold values to 0 and disables the drop threshold.
Step 6	switch(config-if)# [**no**] **l2protocol tunnel shutdown-threshold** [**cdp** | **stp** | **vtp**]	Specifies the maximum number of packets that can be processed on an interface. When the number of packets is exceeded, the port is put in error-disabled state. Optionally, you can specify the Cisco Discovery Protocol (CDP), Spanning Tree Protocol (STP), or VLAN Trunking Protocol (VTP). Valid values for the packets is from 1 to 4096.
Step 7	switch(config-if)# **exit**	Exits interface configuration mode.
Step 8	(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 a threshold for a Layer 2 protocol tunnel port:

```
switch# configure terminal
switch(config)# interface ethernet 7/1
switch(config-if)# switchport
switch(config-if)# switchport mode dot1q-tunnel
switch(config)# l2protocol tunnel drop-threshold 3000
switch(config)# l2protocol tunnel shutdown-threshold 3000
switch(config)# exit
switch# copy running-config startup-config
```

Configuring VLAN Mapping for Selective Q-in-Q on a 802.1Q Tunnel Port

To configure VLAN mapping for selective Q-in-Q on a 802.1Q tunnel port, complete the following steps.

Note You cannot configure one-to-one mapping and selective Q-in-Q on the same interface.

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 mode dot1q-tunnel**	Configure the interface as a tunnel port.
Step 4	switch(config-if)# **switchport vlan mapping** *vlan-id-range* **dot1q-tunnel** *outer vlan-id*	Enters 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 5	switch(config-if)# **exit**	Exits the configuration mode.
Step 6	switch# **show interfaces** *interface-id* **vlan mapping**	Verifies the configuration.
Step 7	switch# **copy running-config startup-config**	(Optional) Saves your entries in the configuration file.

Use the **no switchport vlan mapping** *vlan-id-range* **dot1q-tunnel** *outer vlan-id* command to remove the VLAN mapping configuration.

The following example shows how to drop all VLANs other than the configured mapping and allowed VLANs.

```
switch(config)# interface port-channel201
switch(config-if)# switchport vlan mapping dot1q-tunnel allowed-vlan 201-204
switch(config-if)# switchport vlan mapping 300-400 dot1q-tunnel 500
switch(config-if)# spanning-tree port type edge trunk
switch(config-if)# spanning-tree bpdufilter enable
switch(config-if)# vpc 201
```

Verifying the Q-in-Q Configuration

Use the following command to verify the Q-in-Q tunnel and Layer 2 protocol tunneling configuration information:

Command	Purpose
clear l2protocol tunnel counters [**interface** *if-range*]	Clears all the statistics counters. If no interfaces are specified, the Layer 2 protocol tunnel statistics are cleared for all interfaces.
show dot1q-tunnel [**interface** *if-range*]	Displays a range of interfaces or all interfaces that are in dot1q-tunnel mode.
show l2protocol tunnel [**interface** *if-range* | **vlan** *vlan-id*]	Displays Layer 2 protocol tunnel information for a range of interfaces or all dot1q-tunnel interfaces that are part of a specified VLAN or all interfaces.
show l2protocol tunnel summary	Displays a summary of all ports that have Layer 2 protocol tunnel configurations.
show running-config l2pt	Displays the current Layer 2 protocol tunnel running configuration.

Configuration Example for Q-in-Q and Layer 2 Protocol Tunneling

This example shows a service provider switch that is configured to process Q-in-Q for traffic coming in on Ethernet 7/1. A Layer 2 protocol tunnel is enabled for STP BPDUs. The customer is allocated VLAN 10 (outer VLAN tag).

```
switch# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
switch(config)# vlan 10
switch(config-vlan)# no shutdown
switch(config-vlan)# vlan configuration 8
switch(config-vlan-config)# no ip igmp snooping
switch(config-vlan-config)# exit
switch(config-vlan)# exit
switch(config)# interface ethernet 7/1
switch(config-if)# switchport
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# switchport access vlan 10
switch(config-if)# spanning-tree port type edge
switch(config-if)# l2protocol tunnel stp
switch(config-if)# no shutdown
switch(config-if)# exit
switch(config)# exit
switch#
```

Feature History for Q-in-Q Tunnels and Layer 2 Protocol Tunneling

Table 11: Feature History for Q-in-Q Tunnels and Layer 2 Protocol Tunneling

Feature Name	Release	Feature Information
Q-in-Q VLAN Tunnels	6.0(2)U1(1)	This feature was introduced.
L2 Protocol Tunneling	6.0(2)U1(1)	This feature was introduced.

INDEX

40-Gigabit Ethernet interface speed 8
40-Gigabit Ethernet mode 8
802.1q tunnel port, creating 261
    interfaces 261

A

adding ports 77
    port channels 77
address-family ipv4 unicast 146, 150
address-family ipv6 unicast 146, 150
address-family l2vpn evpn 150, 152–153
advertise 150
associate- vrf 144

B

bandwidth 47
    configuring 47
bud node 116

C

changed information 1
    description 1
channel mode 80
    port channels 80
channel modes 74
    port channels 74
configuration 59
    Layer 3 interfaces 59
        verifying 59
configuration examples 62, 111
    ip tunneling 111
    Layer 3 interfaces 62
configuring 31, 33, 45–49, 84–85
    description parameter 33
    error-disabled recovery interval 31
    interface bandwidth 47
    LACP fast timer rate 84
    LACP port priority 85
    loopback interfaces 49
    routed interfaces 45
    subinterfaces 46
configuring *(continued)*
    VLAN interfaces 48
configuring 10 GbE interface speed 22
configuring 40 GbE interface speed 23
Configuring a DHCP client on an interface 59
configuring an NVE interface 127
configuring LACP 80
configuring rendezvous points 123
Configuring Replication 128
configuring RPs 123
configuring unicast routing protocol 125
configuring VXLAN UDP port 126
creating an NVE interface 127
Creating VXLAN UDP port 126

D

debounce timer 14
    parameters 14
debounce timer, configuring 32
    Ethernet interfaces 32
default interface 14
default settings 44, 102
    ip tunnels 102
    Layer 3 interfaces 44
DHCP client configuration 44
DHCP client configuration limitations 45
DHCP client discovery 44
disabling 25, 28, 31, 33, 239
    CDP 28
    error-disabled recovery 31
    ethernet interfaces 33
    link negotiation 25
    vPCs 239
downlink delay 16

E

enabling 28–30
    CDP 28
    error-disabled detection 29
    error-disabled recovery 30
enabling feature nv overlay 124
enabling PIM 123
enabling VLAN to vn-segment mapping 124

Ethernet interfaces 7, 32
debounce timer, configuring 32
interface speed 7
evpn 150

F

fabric forwarding 144
fabric forwarding anycast-gateway-mac 149
fabric forwarding mode anycast-gateway 149
feature history 65, 88, 112, 266
ip tunnels 112
Layer 3 interfaces 65
port channels 88
q-in-q tunnels, layer 2 protocol tunneling 266
feature nv overlay 145
feature vn-segment 145

G

gre tunnel, configuring 107
interfaces 107
gre tunnels 98
interfaces 98
guidelines 98
ip tunnels 98
guidelines and limitations 43, 231
Layer 3 interfaces 43
vPCs 231
guidelines and limitations for VXLAN 116

H

hardware access-list tcam region arp-ether double-wide 139, 153
host-reachability protocol bgp 144, 149

I

ingress replication 128
interface 149
interface information, displaying 35
layer 2 35
interface MAC address, configuring 55
interface nve 1 153
interface port-channel 83
interface speed 7, 21
configuring 21
Ethernet interfaces 7
interface tunnel 104
interfaces 5–6, 39, 41, 43, 47–49, 54, 61–62, 97–98, 103, 107, 110, 255, 258, 260–262, 265
802.1q tunnel port, creating 261
assigning to a VRF 54
chassis ID 5
configuring bandwidth 47
interfaces *(continued)*
gre tunnel, configuring 107
gre tunnels 98
ip tunnel configuration, verifying 110
ip tunnels 97
ipip tunnel decapsulation-only, configuring 107
ipip tunnel, configuring 107
layer 2 protocol tunnel 262
layer 2 protocol tunnel ports, thresholds configuring 262
layer 2 protocol tunneling 258
Layer 3 39, 61–62
configuration examples 62
monitoring 61
loopback 43, 49
options 5
q-in-q configuration, verifying 265
q-in-q tunneling, guidelines 260
q-in-q tunnels 255
routed 39
tunnel 43
tunnel interface, creating 103
UDLD 6
VLAN 41, 48
configuring 48
ip address 148
ip tunnel configuration, verifying 110
interfaces 110
ip tunneling 111
configuration examples 111
ip tunnels 97–98, 102, 111–112
default settings 102
feature history 112
guidelines 98
interfaces 97
prerequisites 98
standards 111
ipip decapsulate-only 98

L

LACP 68, 73–76, 80–81
configuring 80
marker responders 75
port channel, minlinks 76, 81
port channels 73
system ID 74
LACP fast timer rate 84
configuring 84
lacp max-bundle 83
LACP port priority 85
configuring 85
LACP-enabled vs static 75
port channels 75
layer 2 12, 26, 35
interface information, displaying 35
svi autostate 12

layer 2 *(continued)*
svi autostate, disabling 26
layer 2 mechanism for broadcast, unknown unicast, and multicast traffic 115
layer 2 mechanism for learnt unicast traffic 115
layer 2 protocol tunnel 262
interfaces 262
layer 2 protocol tunneling 258
interfaces 258
Layer 3 interfaces 39, 43–45, 59, 61–62, 65
configuration examples 62
configuring routed interfaces 45
default settings 44
feature history 65
guidelines and limitations 43
interfaces 65
Layer 3 65
feature history 65
MIBs 65
related documents 65
standards 65
MIBs 65
monitoring 61
related documents 65
standards 65
verifying 59
limitations 45
Link Aggregation Control Protocol 68
load balancing 78
port channels 78
configuring 78
loopback interfaces 43, 49
configuring 49

M

mcast-group 149
member vni 144, 149, 153
MIBs 37, 65
Layer 2 interfaces 37
Layer 3 interfaces 65
monitoring 61
Layer 3 interfaces 61
mtu 105
Multi-point IP-in-IP decapsulation 98

N

neighbor 149, 153
new information 1
description 1
no feature nv overlay 154
no feature vn-segment-vlan-based 154
no nv overlay evpn 154
nv overlay evpn 144–145
NVGRE traffic 72

O

overlay-encapsulation vxlan-with-tag 130

P

parameters, about 14
debounce timer 14
physical Ethernet settings 16
point-to-point IP-in-IP encapsulation and decapsulation 98
port channel 86
verifying configuration 86
port channel, minlinks 76, 81
LACP 76, 81
port channeling 68
port channels 47, 67–68, 70, 73, 75–78, 80, 88, 250
adding ports 77
channel mode 80
compatibility requirements 68
configuring bandwidth 47
creating 76
feature history 88
LACP 73
LACP-enabled vs static 75
load balancing 70, 78
port channels 70
moving into a vPC 250
STP 67
port mode 19
interface 19
port modes 9
prerequisites 98
ip tunnels 98

Q

q-in-q configuration, verifying 265
interfaces 265
q-in-q tunneling, guidelines 260
interfaces 260
q-in-q tunnels 255
interfaces 255
q-in-q tunnels, layer 2 protocol 266
feature history 266

R

rd auto 146, 150
related documents 65
Layer 3 interfaces 65
resilient hashing 72
restarting 33
ethernet interfaces 33

retain route-target all 152
route-map permitall out 153
route-map permitall permit 10 152
route-target both auto 146
route-target both auto evpn 146–147
route-target export auto 151
route-target import auto 150
routed interfaces 39, 45, 47
  configuring 45
  configuring bandwidth 47
router bgp 144, 149, 152
router-id 149

S

send-community extended 150, 153
set ip next-hop unchanged 152
SFP+ transceiver 7
show bgp l2vpn evpn 144, 157, 176
show bgp l2vpn evpn summary 144, 175
show interfaces tunnel 105
show ip arp suppression-cache 157
show ip arp suppression-cache detail 175
show l2route evpn fl all 157
show l2route evpn imet all 157
show l2route evpn mac 157
show l2route evpn mac all 177
show l2route evpn mac-ip all 157, 177
show l2route evpn mac-ip all detail 157
show l2route topology 157
show nve peers 175
show nve vni 144, 175
show nve vni summary 144
show nve vrf 157
show running-config interface port-channel 83
show vpc brief 248
show vxlan interface 157, 175
show vxlan interface | count 157
Small form-factor pluggable (plus) transceiver 7
source-interface config 138
source-interface hold-down-time 138
spanning-tree bpdufilter enable 130
standards 65, 111
  ip tunnels 111
  Layer 3 interfaces 65
STP 67
  port channel 67
subinterfaces 40, 46–47
  configuring 46
  configuring bandwidth 47
suppress-arp 144, 153
suppress-mac-route 145
svi autostate 12
  layer 2 12
SVI autostate disable 44
SVI autostate disable, configuring 58
svi autostate, disabling 26
  layer 2 26
switchport access vlan 130
switchport mode dot1q-tunnel 130
symmetric hashing 72

T

tunnel interface 110
  vrf membership, assigning 110
tunnel interfaces 43, 106
  configuring based on PBR 106
tunnel interfaces, creating 103
  interfaces 103
tunnel mode 105
tunnel mode ipip 105
tunnel mode ipv6ipv6 decapsulate-any 105

U

UDLD 6–7
  aggressive mode 7
  defined 6
  nonaggressive mode 7
UDLD modeA 17
  configuring 17
Unidirectional Link Detection 6

V

verifying 59
  Layer 3 interface configuration 59
vlan 146, 148
VLAN 41
  interfaces 41
VLAN interfaces 48
  configuring 48
VLAN to VXLAN VNI mapping 125
vn-segment 146, 148
vni 146, 148, 150
VNI to multicast group mapping 128
vPC terminology 224
vPCs 231, 250
  guidelines and limitations 231
  moving port channels into 250
vrf 150
VRF 54
  assigning an interface to 54
vrf context 144, 146, 148
vrf member 148
vrf membership, assigning 110
  tunnel interface 110