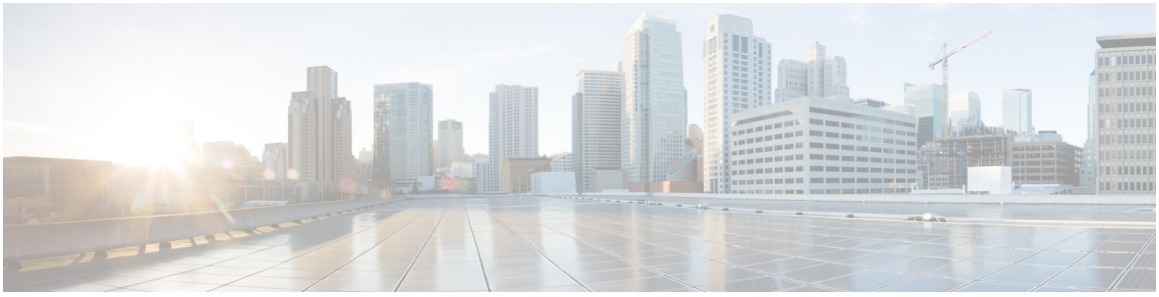
Cisco Nexus 3600 NX-OS Unicast Routing Configuration Guide, Release 7.x

First Published: 2017-08-31

Last Modified: 2020-09-04

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



CONTENTS

PREFACE

Preface xxv

Audience xxv

Document Conventions xxv

Related Documentation for Cisco Nexus 3000 Series Switches xxvi

Documentation Feedback xxvi

Communications, Services, and Additional Information xxvi

CHAPTER 1

New and Changed Information 1

New and Changed Information 1

CHAPTER 2

Overview 3

Licensing Requirements 3

Supported Platforms 3

About Layer 3 Unicast Routing 4

Routing Fundamentals 4

Packet Switching 4

Routing Metrics 5

Path Length 5

Reliability 6

Routing Delay 6

Bandwidth 6

Load 6

Communication Cost 6

Router IDs 6

Autonomous Systems 7

Convergence 8
Load Balancing and Equal Cost Multipath 8
Route Redistribution 8
Administrative Distance 8
Stub Routing 8
Routing Algorithms 10
Static Routes and Dynamic Routing Protocols 10
Interior and Exterior Gateway Protocols 10
Link-State Protocols 10
Layer 3 Virtualization 11
Cisco NX-OS Fowarding Architecture 11
Unicast RIB 11
Adjacency Manager 12
Unicast Forwarding Distribution Module 12
FIB 13
Hardware Forwarding 13
Software Forwarding 13
Summary of Layer 3 Unicast Routing Features 13
IPv4 and IPv6 14
OSPF 14
EIGRP 14
BGP 14
Static Routing 14
Layer 3 Virtualization 14
Route Policy Manager 15
First-Hop Redundancy Protocols 15
Object Tracking 15
Related Topics 15

CHAPTER 3 **Configuring IPv4** 17
About IPv4 17
Multiple IPv4 Addresses 18
Address Resolution Protocol 18
ARP Caching 19

Devices That Do Not Use ARP 19
Reverse ARP 19
Proxy ARP 20
Local Proxy ARP 20
Gratuitous ARP 20
Glean Throttling 20
ICMP 21
ICMP Unreachable Support to Set Source Interface 21
Virtualization Support 21
IPv4 Routes with ECMP 22
Prerequisites for IPv4 22
Guidelines and Limitations 22
Default Settings 22
Configuring IPv4 23
Configuring IPv4 Addressing 23
Configuring Multiple IP Addresses 24
Configuring a Static ARP Entry 25
Configuring Proxy ARP 26
Configuring Local Proxy ARP 26
Configuring Gratuitous ARP 27
Configuring IP Directed Broadcasts 28
Configuring IP Glean Throttling 29
Configuring the Hardware IP Glean Throttle Timeout 30
Configuring the Interface IP Address for the ICMP Source IP Field 30
Configuring Logging for Software Forwarding of IP Packets 31
Verifying the IPv4 Configuration 32
Configuration Examples for IPv4 33

CHAPTER 4

Configuring IPv6 35
About IPv6 35
IPv6 Address Formats 35
IPv6 Unicast Addresses 36
Aggregatable Global Addresses 37
Link-Local Addresses 38

IPv4-Compatible IPv6 Addresses 38
Unique Local Addresses 39
Site-Local Address 40
IPv4 Packet Header 40
Simplified IPv6 Packet Header 40
DNS for IPv6 43
Path MTU Discovery for IPv6 43
CDP IPv6 Address Support 44
ICMP for IPv6 44
IPv6 Neighbor Discovery 45
IPv6 Neighbor Solicitation Message 45
Router Advertisement Message 46
IPv6 Neighbor Redirect Message 47
Virtualization Support 48
IPv6 Routes with ECMP 49
Prerequisites for IPv6 49
Guidelines and Limitations for IPv6 49
Default Settings 49
Configuring IPv6 50
Configuring IPv6 Addressing 50
Configuring IPv6 Neighbor Discovery 52
Optional IPv6 Neighbor Discovery 54
Configuring IPv6 Packet Verification 55
Verifying the IPv6 Configuration 56
Configuration Examples for IPv6 56

CHAPTER 5 **Configuring OSPFv2 57**
Information About OSPFv2 57
Hello Packet 58
Neighbors 58
Adjacency 59
Designated Routers 59
Areas 60
Link-State Advertisements 61

LSA Types 61
Link Cost 62
Flooding and LSA Group Pacing 62
Link-State Database 62
Opaque LSAs 62
OSPFv2 and the Unicast RIB 63
Authentication 63
Simple Password Authentication 63
Cryptographic Authentication 63
MD5 Authentication 63
Advanced Features 64
Stub Area 64
Not-So-Stubby Area 64
Virtual Links 65
Route Redistribution 65
Route Summarization 65
OSPFv2 Stub Router Advertisements 66
Multiple OSPFv2 Instances 66
SPF Optimization 66
BFD 67
Virtualization Support 67
Prerequisites for OSPFv2 67
Guidelines and Limitations 67
Default Settings 67
Configuring Basic OSPFv2 68
Enabling OSPFv2 68
Creating an OSPFv2 Instance 69
Configuring Optional Parameters on an OSPFv2 Instance 70
Configuring Networks in OSPFv2 71
Configuring Authentication for an Area 73
Configuring Authentication for an Interface 75
Configuring Advanced OSPFv2 77
Configuring Filter Lists for Border Routers 77
Configuring Stub Areas 79

Configuring a Totally Stubby Area 80
Configuring NSSA 80
Configuring Virtual Links 82
Configuring Redistribution 84
Limiting the Number of Redistributed Routes 86
Configuring Route Summarization 87
Configuring Stub Route Advertisements 89
Modifying the Default Timers 90
Restarting an OSPFv2 Instance 92
Configuring OSPFv2 with Virtualization 93
Verifying the OSPFv2 Configuration 94
Displaying OSPFv2 Statistics 95
Configuration Examples for OSPFv2 96
Additional References 96
Related Documents 96
MIBs 96

CHAPTER 6 **Configuring OSPFv3** 97
Information About OSPFv3 97
Comparison of OSPFv3 and OSPFv2 98
Hello Packet 98
Neighbors 98
Adjacency 99
Designated Routers 99
Areas 100
Link-State Advertisement 101
LSA Types 101
Link Cost 103
Flooding and LSA Group Pacing 103
Link-State Database 103
Multi-Area Adjacency 104
OSPFv3 and the IPv6 Unicast RIB 104
Address Family Support 104
Advanced Features 104

Stub Area 105
Not-So-Stubby Area 105
Virtual Links 105
Route Redistribution 106
Route Summarization 106
High Availability and Graceful Restart 107
Multiple OSPFv3 Instances 108
SPF Optimization 108
BFD 108
Virtualization Support 108
Prerequisites for OSPFv3 108
Guidelines and Limitations for OSPFv3 109
Default Settings 109
Configuring Basic OSPFv3 110
Enabling OSPFv3 110
Creating an OSPFv3 Instance 111
Configuring Networks in OSPFv3 113
Configuring Advanced OSPFv3 115
Configuring Filter Lists for Border Routers 115
Configuring Stub Areas 117
Configuring a Totally Stubby Area 118
Configuring NSSA 118
Configuring Multi-Area Adjacency 120
Configuring Virtual Links 121
Configuring Redistribution 123
Limiting the Number of Redistributed Routes 125
Configuring Route Summarization 127
Modifying the Default Timers 128
Configuring Graceful Restart 130
Restarting an OSPFv3 Instance 132
Configuring OSPFv3 with Virtualization 132
Verifying the OSPFv3 Configuration 134
Monitoring OSPFv3 135
Configuration Examples for OSPFv3 135

Related Topics 135
Additional References 136
MIBs 136

CHAPTER 7 Configuring EIGRP 137
Information About EIGRP 137
EIGRP Components 137
Reliable Transport Protocol 138
Neighbor Discovery and Recovery 138
Diffusing Update Algorithm 138
EIGRP Route Updates 139
Internal Route Metrics 139
External Route Metrics 140
EIGRP and the Unicast RIB 140
Advanced EIGRP 140
Address Families 140
Authentication 141
Stub Routers 141
Route Summarization 141
Route Redistribution 142
Load Balancing 142
Split Horizon 142
Virtualization Support 142
Prerequisites for EIGRP 143
Guidelines and Limitations 143
Default Settings 143
Configuring Basic EIGRP 144
Enabling the EIGRP Feature 144
Creating an EIGRP Instance 145
Restarting an EIGRP Instance 147
Shutting Down an EIGRP Instance 148
Configuring Advanced EIGRP 148
Configuring Authentication in EIGRP 148
Configuring EIGRP Stub Routing 150

Configuring a Summary Address for EIGRP 151
Redistributing Routes into EIGRP 151
Limiting the Number of Redistributed Routes 153
Configuring Load Balancing in EIGRP 155
Adjusting the Interval Between Hello Packets and the Hold Time 156
Disabling Split Horizon 156
Tuning EIGRP 157
Configuring Virtualization for EIGRP 159
Verifying the EIGRP Configuration 161
Displaying EIGRP Statistics 161
Configuration Examples for EIGRP 161
Related Topics 162
Additional References 162
MIBs 162

CHAPTER 8

Configuring IS-IS 163
About IS-IS 163
IS-IS Overview 163
IS-IS Areas 164
NET and System ID 164
Designated Intermediate System 165
IS-IS Authentication 165
Mesh Groups 165
Overload Bit 165
Route Summarization 166
Configuring Redistribution 166
Load Balancing 167
BFD 168
Virtualization Support 168
High Availability and Graceful Restart 168
Multiple IS-IS Instances 169
Prerequisites for IS-IS 169
Guidelines and Limitations for IS-IS 169
Default Settings 169

Configuring IS-IS 170
IS-IS Configuration Modes 170
Router Configuration Mode 170
Router Address Family Configuration Mode 170
Enabling the IS-IS Feature 170
Creating an IS-IS Instance 171
Restarting an IS-IS Instance 173
Shutting Down IS-IS 174
Configuring IS-IS Authentication in an Area 174
Configuring IS-IS Authentication on an Interface 175
Configuring a Mesh Group 176
Configuring a Designated Intermediate System 177
Configuring Dynamic Host Exchange 177
Setting the Overload Bit 177
Configuring the Attached Bit 178
Configuring the Transient Mode for Hello Padding 178
Configuring a Summary Address 178
Configuring Redistribution 180
Limiting the Number of Redistributed Routes 181
Disabling Strict Adjacency Mode 183
Configuring a Graceful Restart 184
Configuring Virtualization 185
Tuning IS-IS 187
Verifying the IS-IS Configuration 189
Monitoring IS-IS 190
Configuration Examples for IS-IS 191
Related Topics 191

CHAPTER 9 **Configuring Basic BGP 193**
About Basic BGP 193
BGP Autonomous Systems 194
4-Byte AS Number Support 194
Administrative Distance 194
BGP Peers 194

BGP Sessions 194
Dynamic AS Numbers for Prefix Peers 195
BGP Router Identifier 195
BGP Path Selection 195
Step 1—Comparing Pairs of Paths 196
Step 2—Determining the Order of Comparisons 198
Step 3—Determining the Best-Path Change Suppression 198
BGP and the Unicast RIB 198
BGP Virtualization 199
Prerequisites for Basic BGP 199
Guidelines and Limitations for BGP 199
CLI Configuration Modes 200
Global Configuration Mode 200
Address Family Configuration Mode 200
Neighbor Configuration Mode 200
Neighbor Address Family Configuration Mode 201
Default Settings 201
Configuring Basic BGP 202
Enabling the BGP Feature 202
Creating a BGP Instance 203
Restarting a BGP Instance 204
Shutting Down BGP 204
Configuring BGP Peers 205
Distributing the Default Static Route to All BGP VRFs 207
Configuring Update Announcement Delay Timers 208
Configuring BGP Reconnect Interval 209
Configuring Dynamic AS Numbers for Prefix Peers 210
Clearing BGP Information 211
Verifying the Basic BGP Configuration 213
Displaying BGP Statistics 215
Configuration Examples for Basic BGP 215
Related Topics 215
Where to Go Next 216
Additional References 216

MIBs 216

CHAPTER 10 **Configuring Advanced BGP** 217

Information About Advanced BGP 217

Peer Templates 217

Authentication 218

Route Policies and Resetting BGP Sessions 218

eBGP 219

eBGP Next-Hop Unchanged 219

iBGP 219

AS Confederations 220

Route Reflector 220

Capabilities Negotiation 221

Route Dampening 221

Load Sharing and Multipath 222

Route Aggregation 222

BGP Conditional Advertisement 223

BGP Next-Hop Address Tracking 223

Site of Origin 224

Route Redistribution 224

BFD 224

Tuning BGP 225

BGP Timers 225

Tuning the Best-Path Algorithm 225

Multiprotocol BGP 225

RFC 5549 225

Prerequisites for Advanced BGP 225

Guidelines and Limitations for Advanced BGP 226

Default Settings for BGP 227

Configuring Advanced BGP 227

Enabling IP Forward on an Interface 227

Configuring BGP Session Templates 228

Configuring BGP Peer-Policy Templates 230

Configuring BGP Peer Templates 232

Configuring Prefix Peering 235
Configuring BGP Authentication 236
Resetting a BGP Session 236
Modifying the Next-Hop Address 237
Configuring BGP Next-Hop Address Tracking 237
Configuring Next-Hop Filtering 238
Controlling Reflected Routes Through Next-Hop-Self 238
Shrinking Next-Hop Groups When A Session Goes Down 239
Disabling Capabilities Negotiation 239
Configuring BGP Additional Paths 239
Advertising the Capability of Sending and Receiving Additional Paths 239
Configuring the Sending and Receiving of Additional Paths 240
Configuring Advertised Paths 241
Configuring Additional Path Selection 242
Configuring eBGP 242
Configuring eBGP Next-Hop Unchanged 243
Disabling eBGP Single-Hop Checking 243
Configuring eBGP Multihop 244
Configuring eBGP Routes in the Same Autonomous System 244
Disabling a Fast External Failover 245
Limiting the AS-path Attribute 245
Configuring Local AS Support 246
Configuring AS Confederations 246
Configuring an Autonomous System Path Containing Your Own Autonomous System 247
Configuring Route Reflector 248
Configuring Route Dampening 250
Configuring Load Sharing and ECMP 250
Configuring Maximum Prefixes 251
Configuring Dynamic Capability 251
Configuring Aggregate Addresses 251
Suppressing BGP Routes 252
Configuring BGP Conditional Advertisement 252
Configuring Route Redistribution 254
Disabling BGP Dampening with Redistribution 256

Configuring Multiprotocol BGP 256
Configuring BGP Extended Community Site of Origin 257
Tuning BGP 258
Configuring Virtualization 258
Verifying the Advanced BGP Configuration 260
Displaying BGP Statistics 261
Related Topics 262
Additional References 262
MIBs 262

CHAPTER 11 **Configuring Static Routing** 263
About Static Routing 263
Administrative Distance 263
Directly Connected Static Routes 264
Fully Specified Static Routes 264
Floating Static Routes 264
Remote Next-hops for Static Routes 264
BFD 264
Virtualization Support 265
Prerequisites for Static Routing 265
Guidelines and Limitations 265
Default Settings 265
Configuring Static Routing 265
Configuring a Static Route 265
Configuring Virtualization 266
Verifying the Static Routing Configuration 267
Configuration Examples for Static Routing 267

CHAPTER 12 **Configuring Layer 3 Virtualization** 269
About Layer 3 Virtualization 269
Overview of Layer 3 Virtualization 269
VRF and Routing 270
VRF-Lite 270
VRF-Aware Services 270

Reachability 271
Filtering 271
Combining Reachability and Filtering 272
Guidelines and Limitations for VRF 272
Guidelines and Limitations for VRF-Lite 272
Guidelines and Limitations for VRF Route Leaking 273
Default Settings 273
Configuring VRFs 274
Creating a VRF 274
Assigning VRF Membership to an Interface 275
Configuring VRF Parameters for a Routing Protocol 276
Configuring a VRF-Aware Service 277
Setting the VRF Scope 278
Verifying the VRF Configuration 279
Configuration Examples for VRFs 279

CHAPTER 13 **Configuring the Unicast RIB and FIB 283**
About the Unicast RIB and FIB 283
Layer 3 Consistency Checker 284
FIB Tables 284
Virtualization Support 285
Managing the Unicast RIB and FIB 285
Displaying Module FIB Information 285
Configuring Load Sharing in the Unicast FIB 286
Displaying Routing and Adjacency Information 288
Triggering the Layer 3 Consistency Checker 289
Clearing Forwarding Information in the FIB 290
Estimating Memory Requirements for Routes 290
Clearing Routes in the Unicast RIB 291
Verifying the Unicast RIB and FIB Configuration 292

CHAPTER 14 **Configuring Route Policy Manager 293**
About Route Policy Manager 293
Prefix Lists 293

Prefix List Masks 294
MAC Lists 294
Route Maps 294
Match Criteria 295
Set Changes 295
Access Lists 295
AS Numbers for BGP 295
AS-Path Lists for BGP 296
Community Lists for BGP 296
Extended Community Lists for BGP 296
Route Redistribution and Route Maps 297
Guidelines and Limitations for Route Policy Manager 297
Default Settings 298
Configuring Route Policy Manager 298
Configuring IP Prefix Lists 298
Configuring MAC Lists 300
Configuring AS-Path Lists 300
Configuring Community Lists 301
Configuring Route Maps 303
Verifying the Route Policy Manager Configuration 308
Related Topics 308

CHAPTER 15 **Configuring Bidirectional Forwarding Detection** 309
Information About BFD 309
Asynchronous Mode 309
BFD Detection of Failures 310
BFD Echo Function 310
Security 310
Virtualization Support 310
Prerequisites for BFD 311
Guidelines and Limitations 311
Default Settings 312
Configuring BFD 313
Configuration Hierarchy 313

Task Flow for Configuring BFD 313
Enabling the BFD Feature 313
Configuring Global BFD Parameters 314
Configuring BFD on an Interface 315
Configuring BFD on a Port Channel 316
Configuring the BFD Echo Function 317
Configuring BFD on BGP 319
Configuring BFD on PIM 319
Configuring BFD on OSPFv2 320
Configuring BFD for Static Routes 321
Configuring BFD for IPv6 322
Configuring Global BFD Parameters for IPv6 322
Configuring Per Interface BFD Parameters for IPv6 323
Configuring BFD for IPv6 on OSPFv3 323
Configuring BFD on IPv6 Static Routes 324
Configuring BFD Echo Mode 325
Configuring BFD Session Echo Interval 326
Configuring a BFD Echo Interface 326
Configuring the BFD Slow Timer 327
Verifying the BFD Configuration 327
Monitoring BFD 328

CHAPTER 16 **Configuring Policy-Based Routing** 329
About Policy-Based Routing 329
Policy Route Maps 329
Set Criteria for Policy-Based Routing 330
Route-Map Processing Logic 330
Policy-Based Routing Filtering Options 331
Prerequisites for Policy-Based Routing 331
Guidelines and Limitations for Policy-Based Routing 331
Default Settings 332
Configuring Policy-Based Routing 332
Enabling the Policy-Based Routing Feature 332
Configuring a Route Policy 333

Verifying the Policy-Based Routing Configuration 334
Configuration Examples for Policy Based-Routing 334

CHAPTER 17 Configuring HSRP 337
Information About HSRP 337
HSRP Overview 337
HSRP for IPv4 338
HSRP Versions 339
HSRP Subnet VIP 339
HSRP Authentication 340
HSRP Messages 340
HSRP Load Sharing 340
Object Tracking and HSRP 341
Virtualization Support 341
Prerequisites for HSRP 341
Guidelines and Limitations for HSRP 341
Default Settings for HSRP 342
Configuring HSRP 343
Enabling the HSRP Feature 343
Configuring the HSRP Version 343
Configuring an HSRP Group for IPv4 343
Configuring the HSRP Virtual MAC Address 345
Authenticating HSRP 346
Configuring HSRP Object Tracking 348
Configuring the HSRP Priority 350
Customizing HSRP 351
Verifying the HSRP Configuration 352
Configuration Examples for HSRP 352
Additional References 353
Related Documents 353
MIBs 353

CHAPTER 18 Configuring VRRP 355
Information About VRRP 355

VRRP Operation 355
VRRP Benefits 357
Multiple VRRP Groups 357
VRRP Router Priority and Preemption 358
VRRP Advertisements 359
VRRP Authentication 359
VRRPv3 359
Virtualization Support 359
Guidelines and Limitations for VRRP 359
Default Settings for VRRP 360
Configuring VRRP 361
Enabling the VRRP Feature 361
Configuring VRRP Groups 361
Configuring VRRP Priority 362
Configuring VRRP Authentication 364
Configuring Time Intervals for Advertisement Packets 365
Disabling Preemption 366
Configuring VRRP Interface State Tracking 368
Configuring VRRPv3 369
Enabling VRRPv3 369
Configuring VRRPv3 Groups 369
Configuring the Delay Period for FHRP Client Initialization 372
Configuring VRRPv3 Control Groups 372
Verifying the VRRPv2 Configuration 373
Verifying the VRRPv3 Configuration 374
Displaying VRRP Statistics 374
Configuration Examples for VRRPv2 375
Configuration Example for VRRPv3 376
Additional References 376
Related Documents 377

CHAPTER 19 **Configuring Object Tracking** 379
Information About Object Tracking 379
Object Tracking Overview 379

Object Track List 380
Virtualization Support 380
Guidelines and Limitations for Object Tracking 380
Default Settings for Object Tracking 381
Configuring Object Tracking 381
Configuring Object Tracking for an Interface 381
Configuring Object Tracking for Route Reachability 382
Configuring an Object Track List with a Boolean Expression 383
Configuring an Object Track List with a Percentage Threshold 384
Configuring an Object Track List with a Weight Threshold 385
Configuring an Object Tracking Delay 386
Configuring Object Tracking for a Nondefault VRF 388
Verifying the Object Tracking Configuration 389
Configuration Examples for Object Tracking 389
Related Topics 389
Additional References 390
Related Documents 390
IETF RFCs 390

APPENDIX A IETF RFCs 391
IETF RFCs 391

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.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

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: https://www.cisco.com/c/en/us/about/legal/trademarks.html. 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. (1721R)

© 2017–2020 Cisco Systems, Inc. All rights reserved.

Preface

This preface includes the following sections:

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

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.

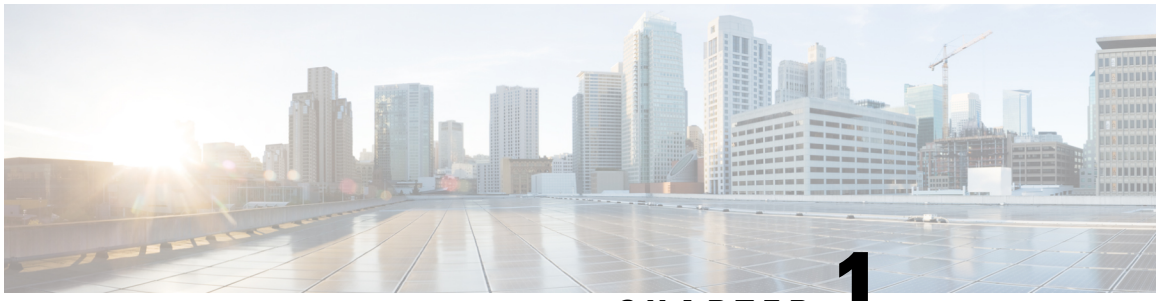
Communications, Services, and Additional Information

- 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 provides release-specific information for new and changed feature in the Cisco Nexus 3600 NX-OS Unicast Routing Configuration Guide, Release 7.x. The latest version of this document is available at the following Cisco website:

https://www.cisco.com/c/en/us/support/switches/nexus-3000-series-switches/products-installation-and-configuration-guides-list.html

To check for additional information, see the Cisco Nexus 3600 NX-OS Release Notes, Release 7.x available at the following Cisco website:

https://www.cisco.com/c/en/us/support/switches/nexus-3000-series-switches/products-release-notes-list.html

- New and Changed Information, on page 1

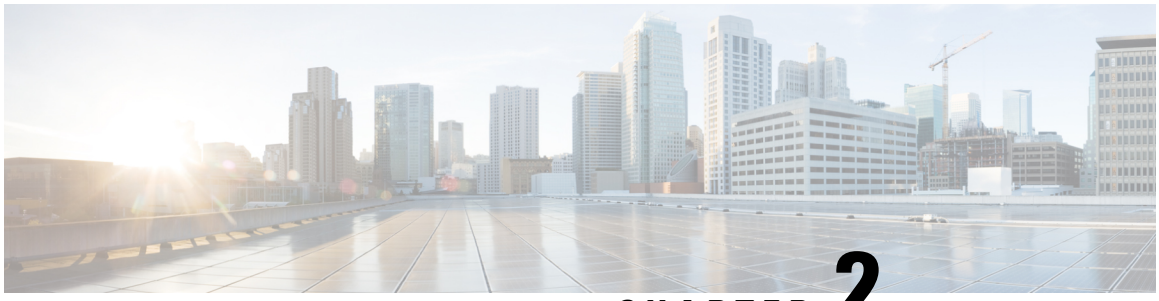
New and Changed Information

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

Table 1: New and Changed Feature Information

Feature	Description	Changed in Release	Where Documented
BGP Routing	For **next-hop-self**, added the optional **all** argument which controls whether reflected routes have their next hops changed.	7.0(3)I7(6)	Controlling Reflected Routes Through Next Hop Self
HSRP	Added the ability to configure an HSRP subnet virtual IP address in a different subnet than that of the SVI IP address.	7.0(3)F3(3)	Configuring HSRP
Policy-based routing (PBR)	Introduced this feature.	7.0(3)F3(3)	Configuring Policy-Based Routing
Unicast routing	First release.	7.0(3)F3(1)	

New and Changed Information



CHAPTER 2

Overview

This chapter introduces the basic concepts for Layer 3 unicast routing protocols in Cisco Nexus 3600 platform switches.

This chapter includes the following sections:

- Licensing Requirements, on page 3
- Supported Platforms, on page 3
- About Layer 3 Unicast Routing, on page 4
- Routing Algorithms, on page 10
- Layer 3 Virtualization, on page 11
- Cisco NX-OS Fowarding Architecture, on page 11
- Unicast RIB, on page 11
- Adjacency Manager, on page 12
- Unicast Forwarding Distribution Module, on page 12
- FIB, on page 13
- Hardware Forwarding, on page 13
- Software Forwarding, on page 13
- Summary of Layer 3 Unicast Routing Features, on page 13
- First-Hop Redundancy Protocols, on page 15
- Object Tracking, on page 15
- Related Topics, on page 15

Licensing Requirements

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

Supported Platforms

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

About Layer 3 Unicast Routing

Layer 3 unicast routing involves two basic activities: determining optimal routing paths and packet switching. You can use routing algorithms to calculate the optimal path from the router to a destination. This calculation depends on the algorithm selected, route metrics, and other considerations such as load balancing and alternate path discovery.

Routing Fundamentals

Routing protocols use a metric to evaluate the best path to the destination. A metric is a standard of measurement, such as a path bandwidth, that routing algorithms use to determine the optimal path to a destination. To aid path determination, routing algorithms initialize and maintain routing tables, that contain route information such as the IP destination address and the address of the next router or next-hop. Destination and next-hop associations tell a router that an IP destination can be reached optimally by sending the packet to a particular router that represents the next-hop on the way to the final destination. When a router receives an incoming packet, it checks the destination address and attempts to associate this address with the next-hop. See the Unicast RIB section for more information about the route table.

Routing tables can contain other information such as the data about the desirability of a path. Routers compare metrics to determine optimal routes, and these metrics differ depending on the design of the routing algorithm used. See the Routing Metrics section.

Routers communicate with one another and maintain their routing tables by transmitting a variety of messages. The routing update message is one of these messages that consists of all or a portion of a routing table. By analyzing routing updates from all other routers, a router can build a detailed picture of the network topology. A link-state advertisement, which is another example of a message sent between routers, informs other routers of the link state of the sending router. You can also use link information to enable routers to determine optimal routes to network destinations. For more information, see the Routing Algorithms section.

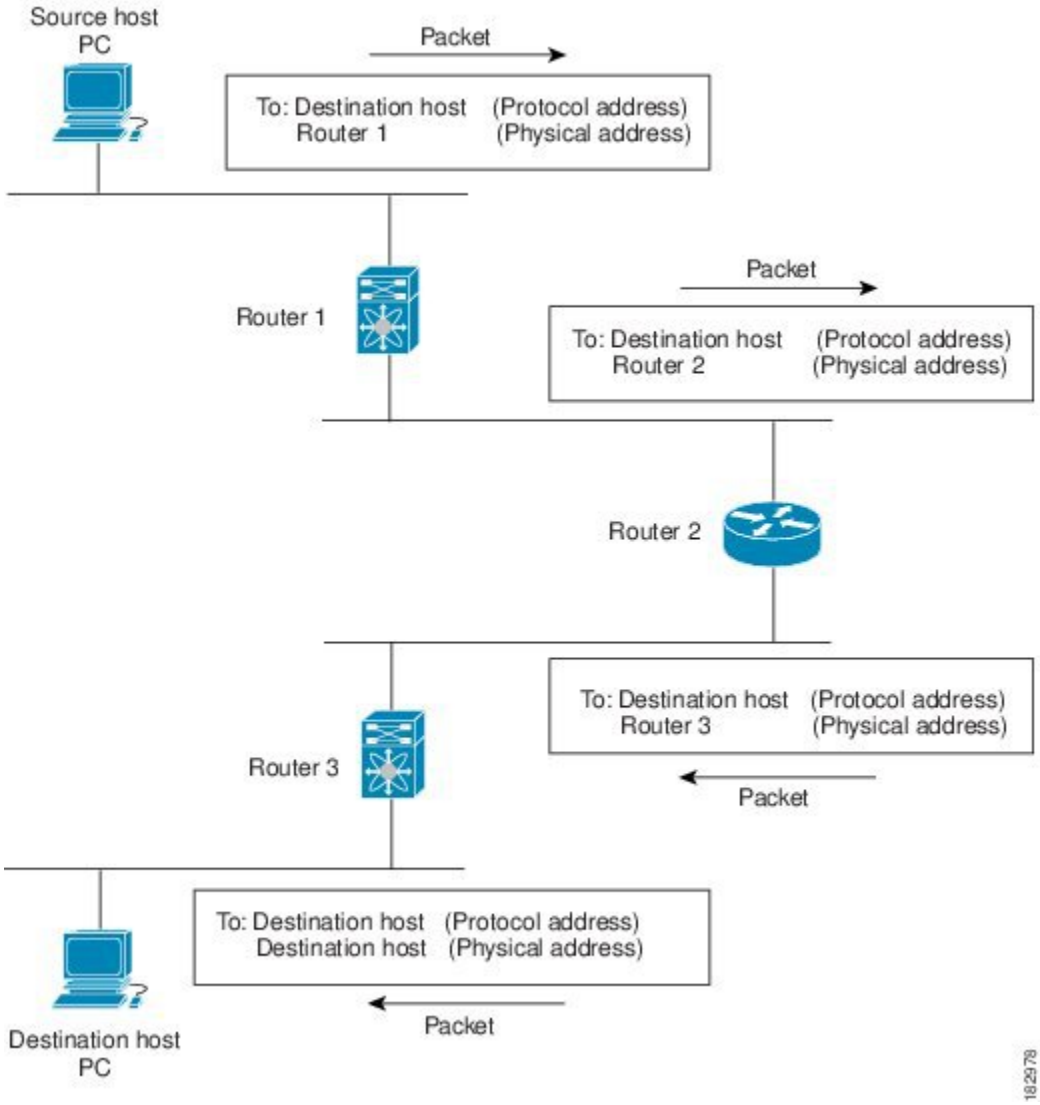
Packet Switching

In packet switching, a host determines that it must send a packet to another host. Having acquired a router address by some means, the source host sends a packet addressed specifically to the router physical (Media Access Control [MAC]-layer) address but with the IP (network layer) address of the destination host.

The router examines the destination IP address and tries to find the IP address in the routing table. If the router does not know how to forward the packet, it typically drops the packet. If the router knows how to forward the packet, it changes the destination MAC address to the MAC address of the next-hop router and transmits the packet.

The next-hop might be the ultimate destination host or another router that executes the same switching decision process. As the packet moves through the internetwork, its physical address changes, but its protocol address remains constant (see the following figure).

Figure 1: Packet Header Updates Through a Network



Routing Metrics

Routing algorithms use many different metrics to determine the best route. Sophisticated routing algorithms can base route selection on multiple metrics

Path Length

The path length is the most common routing metric. Some routing protocols allow you to assign arbitrary costs to each network link. In this case, the path length is the sum of the costs associated with each link traversed. Other routing protocols define hop count, a metric that specifies the number of passes through internetworking products, such as routers, that a packet must take from a source to a destination.

Reliability

The reliability, in the context of routing algorithms, is the dependability (in terms of the bit-error rate) of each network link. Some network links might go down more often than others. After a network fails, certain network links might be repaired more easily or more quickly than other links. The reliability factors that you can take into account when assigning the reliability rating are arbitrary numeric values that you usually assign to network links.

Routing Delay

The routing delay is the length of time required to move a packet from a source to a destination through the internetwork. The delay depends on many factors, including the bandwidth of intermediate network links, the port queues at each router along the way, the network congestion on all intermediate network links, and the physical distance that the packet needs to travel. Because the routing delay is a combination of several important variables, it is a common and useful metric.

Bandwidth

The bandwidth is the available traffic capacity of a link. For example, a 10-Gigabit Ethernet link would be preferable to a 1-Gigabit Ethernet link. Although the bandwidth is the maximum attainable throughput on a link, routes through links with greater bandwidth do not necessarily provide better routes than routes through slower links. For example, if a faster link is busier, the actual time required to send a packet to the destination could be greater.

Load

The load is the degree to which a network resource, such as a router, is busy. You can calculate the load in a variety of ways, including CPU utilization and packets processed per second. Monitoring these parameters on a continual basis can be resource intensive.

Communication Cost

The communication cost is a measure of the operating cost to route over a link. The communication cost is another important metric, especially if you do not care about performance as much as operating expenditures. For example, the line delay for a private line might be longer than a public line, but you can send packets over your private line rather than through the public lines that cost money for usage time.

Router IDs

Each routing process has an associated router ID. You can configure the router ID to any interface in the system. If you do not configure the router ID, Cisco NX-OS selects the router ID based on the following criteria:

- Cisco NX-OS prefers loopback0 over any other interface. If loopback0 does not exist, Cisco NX-OS prefers the first loopback interface over any other interface type.
- If you have not configured any loopback interfaces, Cisco NX-OS uses the first interface in the configuration file as the router ID. If you configure any loopback interface after Cisco NX-OS selects the router ID, the loopback interface becomes the router ID. If the loopback interface is not loopback0 and you configure loopback0 later with an IP address, the router ID changes to the IP address of loopback0.
- If the interface that the router ID is based on changes, that new IP address becomes the router ID. If any other interface changes its IP address, there is no router ID change.

Autonomous Systems

An autonomous system (AS) is a network controlled by a single technical administration entity. Autonomous systems divide global external networks into individual routing domains, where local routing policies are applied. This organization simplifies routing domain administration and simplifies consistent policy configuration.

Each autonomous system can support multiple interior routing protocols that dynamically exchange routing information through route redistribution. The Regional Internet Registries assign a unique number to each public autonomous system that directly connects to the Internet. This autonomous system number (AS number) identifies both the routing process and the autonomous system.

Cisco NX-OS supports 4-byte AS numbers. The following table lists the AS number ranges.

Table 2: Table 1-1 AS Numbers

2-Byte Numbers	4-Byte Numbers in AS.dot Notation	4-Byte Numbers in plaintext Notation	Purpose
1 to 64511	0.1 to 0.64511	1 to 64511	Public AS (assigned by RIR) **Note** RIR = Regional Internet Registries
64512 to 65534	0.64512 to 0.65534	64512 to 65534	Private AS (assigned by local administrator)
65535	0.65535	65535	Reserved
N/A	1.0 to 65535.65535	65536 to 4294967295	Public AS (assigned by RIR) **Note** RIR = Regional Internet Registries

Private autonomous system numbers are used for internal routing domains but must be translated by the router for traffic that is routed out to the Internet. You should not configure routing protocols to advertise private autonomous system numbers to external networks. By default, Cisco NX-OS does not remove private autonomous system numbers from routing updates.

Note The autonomous system number assignment for public and private networks is governed by the Internet Assigned Number Authority (IANA). For information about autonomous system numbers, including the reserved number assignment, or to apply to register an autonomous system number, refer to the following URL:

http://www.iana.org/

Convergence

A key aspect to measure for any routing algorithm is how much time a router takes to react to network topology changes. When a part of the network changes for any reason, such as a link failure, the routing information in different routers might not match. Some routers will have updated information about the changed topology, other routers will still have the old information. The convergence is the amount of time before all routers in the network have updated, matching routing information. The convergence time varies depending on the routing algorithm. Fast convergence minimizes the chance of lost packets caused by inaccurate routing information.

Load Balancing and Equal Cost Multipath

Routing protocols can use load balancing or equal cost multipath (ECMP) to share traffic across multiple paths.When a router learns multiple routes to a specific network, it installs the route with the lowest administrative distance in the routing table. If the router receives and installs multiple paths with the same administrative distance and cost to a destination, load balancing can occur. Load balancing distributes the traffic across all the paths, sharing the load. The number of paths used is limited by the number of entries that the routing protocol puts in the routing table. Cisco NX-OS supports up to 16 paths to a destination.

The Enhanced Interior Gateway Routing Protocol (EIGRP) also supports unequal cost load balancing. For more information, see Configuring EIGRP.

Route Redistribution

If you have multiple routing protocols configured in your network, you can configure these protocols to share routing information by configuring route redistribution in each protocol. For example, you can configure Open Shortest Path First (OSPF) to advertise routes learned from the Border Gateway Protocol (BGP). You can also redistribute static routes into any dynamic routing protocol. The router that is redistributing routes from another protocol sets a fixed route metric for those redistributed routes and avoids the problem of incompatible route metrics between the different routing protocols. For example, routes redistributed from EIGRP into OSPF are assigned a fixed link cost metric that OSPF understands.

Route redistribution also uses an administrative distance (see the Administrative Distance section) to distinguish between routes learned from two different routing protocols. The preferred routing protocol is given a lower administrative distance so that its routes are chosen over routes from another protocol with a higher administrative distance assigned.

Administrative Distance

An administrative distance is a rating of the trustworthiness of a routing information source. A higher value means a lower trust rating. Typically, a route can be learned through more than one protocol. Administrative distance is used to discriminate between routes learned from more than one protocol. The route with the lowest administrative distance is installed in the IP routing table.

Stub Routing

You can use stub routing in a hub-and-spoke network topology, where one or more end (stub) networks are connected to a remote router (the spoke) that is connected to one or more distribution routers (the hub). The remote router is adjacent only to one or more distribution routers. The only route for IP traffic to follow into the remote router is through a distribution router. This type of configuration is commonly used in WAN

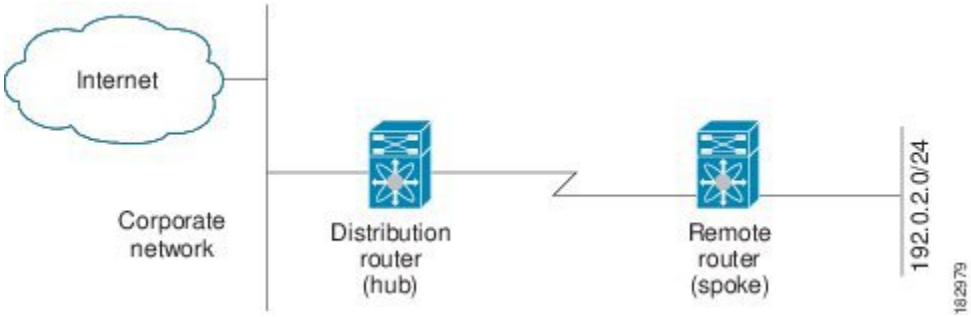
topologies in which the distribution router is directly connected to a WAN. The distribution router can be connected to many more remote routers. Often, the distribution router is connected to 100 or more remote routers. In a hub-and-spoke topology, the remote router must forward all nonlocal traffic to a distribution router, so it becomes unnecessary for the remote router to hold a complete routing table. Generally, the distribution router sends only a default route to the remote router.

Only specified routes are propagated from the remote (stub) router. The stub router responds to all queries for summaries, connected routes, redistributed static routes, external routes, and internal routes with the message "inaccessible." A router that is configured as a stub sends a special peer information packet to all neighboring routers to report its status as a stub router.

Any neighbor that receives a packet informing it of the stub status does not query the stub router for any routes, and a router that has a stub peer does not query that peer. The stub router depends on the distribution router to send the proper updates to all peers.

The following figure shows a simple hub-and-spoke configuration.

Figure 2: Simple Hub-and-Spoke Network



Stub routing does not prevent routes from being advertised to the remote router. Figure **Simple Hub-and-Spoke Network** shows that the remote router can access the corporate network and the Internet through the distribution router only. A full route table on the remote router, in this example, serves no functional purpose because the path to the corporate network and the Internet would always be through the distribution router. A larger route table would reduce only the amount of memory required by the remote router. The bandwidth and memory used can be lessened by summarizing and filtering routes in the distribution router. In this network topology, the remote router does not need to receive routes that have been learned from other networks because the remote router must send all nonlocal traffic, regardless of its destination, to the distribution router. To configure a true stub network, you should configure the distribution router to send only a default route to the remote router.

OSPF supports stub areas and EIGRP supports stub routers.

Note The EIGRP stub routing feature should be used only on stub devices. A stub device is defined as a device connected to the network core or distribution layer through which core transit traffic should not flow. The only route for IP traffic to follow into the remote router is through a distribution router. A stub device should not have any EIGRP neighbors other than distribution devices. Ignoring this restriction will cause undesirable behavior.

Routing Algorithms

Routing algorithms determine how a router gathers and reports reachability information, how it deals with topology changes, and how it determines the optimal route to a destination. Various types of routing algorithms exist, and each algorithm has a different impact on network and router resources. Routing algorithms use a variety of metrics that affect calculation of optimal routes. You can classify routing algorithms by type, such as static or dynamic, and interior or exterior.

Static Routes and Dynamic Routing Protocols

Static routes are route table entries that you manually configure. These static routes do not change unless you reconfigure them. Static routes are simple to design and work well in environments where network traffic is relatively predictable and where network design is relatively simple.

Because static routing systems cannot react to network changes, you should not use them for today's large, constantly changing networks. Most routing protocols today use dynamic routing algorithms, which adjust to changing network circumstances by analyzing incoming routing update messages. If the message indicates that a network change has occurred, the routing software recalculates routes and sends out new routing update messages. These messages permeate the network, triggering routers to rerun their algorithms and change their routing tables accordingly.

You can supplement dynamic routing algorithms with static routes where appropriate. For example, you should configure each subnetwork with a static route to the IP default gateway or router of last resort (a router to which all unrouteable packets are sent).

Interior and Exterior Gateway Protocols

You can separate networks into unique routing domains or autonomous systems. An autonomous system is a portion of an internetwork under common administrative authority that is regulated by a particular set of administrative guidelines. Routing protocols that route between autonomous systems are called exterior gateway protocols or interdomain protocols. BGP is an example of an exterior gateway protocol. Routing protocols used within an autonomous system are called interior gateway protocols or intradomain protocols. EIGRP and OSPF are examples of interior gateway protocols.

Link-State Protocols

The link-state protocols, also known as shortest path first (SPF), share information with neighboring routers. Each router builds a link-state advertisement (LSA), which contains information about each link and directly connected neighbor router.

Each LSA has a sequence number. When a router receives an LSA and updates its link-state database, the LSA is flooded to all adjacent neighbors. If a router receives two LSAs with the same sequence number (from the same router), the router does not flood the last LSA received to its neighbors to prevent an LSA update loop. Because the router floods the LSAs immediately after they receive them, convergence time for link-state protocols is minimized.

Discovering neighbors and establishing adjacency is an important part of a link state protocol. Neighbors are discovered using special Hello packets that also serve as keepalive notifications to each neighbor router. Adjacency is the establishment of a common set of operating parameters for the link-state protocol between neighbor routers.

The LSAs received by a router are added to its link-state database. Each entry consists of the following parameters:

- Router ID (for the router that originated the LSA)
- Neighbor ID
- Link cost
- Sequence number of the LSA
- Age of the LSA entry

The router runs the SPF algorithm on the link-state database, building the shortest path tree for that router. This SPF tree is used to populate the routing table.

In link-state algorithms, each router builds a picture of the entire network in its routing tables. The link-state algorithms send small updates everywhere, while distance vector algorithms send larger updates only to neighboring routers.

Because they converge more quickly, link-state algorithms are somewhat less prone to routing loops than distance vector algorithms. However, link-state algorithms require more CPU power and memory than distance vector algorithms. Link-state algorithms can be more expensive to implement and support. Link-state protocols are generally more scalable than distance vector protocols.

OSPF is an example of a link-state protocol.

Layer 3 Virtualization

Cisco NX-OS supports multiple Virtual Routing and Forwarding (VRF) instances and multiple routing information bases (RIBs) to support multiple address domains. Each VRF is associated with a RIB and this information is collected by the forwarding information base (FIB). A VRF represents a Layer 3 addressing domain. Each Layer 3 interface (logical or physical) belongs to one VRF. For more information, see Configuring Layer 3 Virtualization.

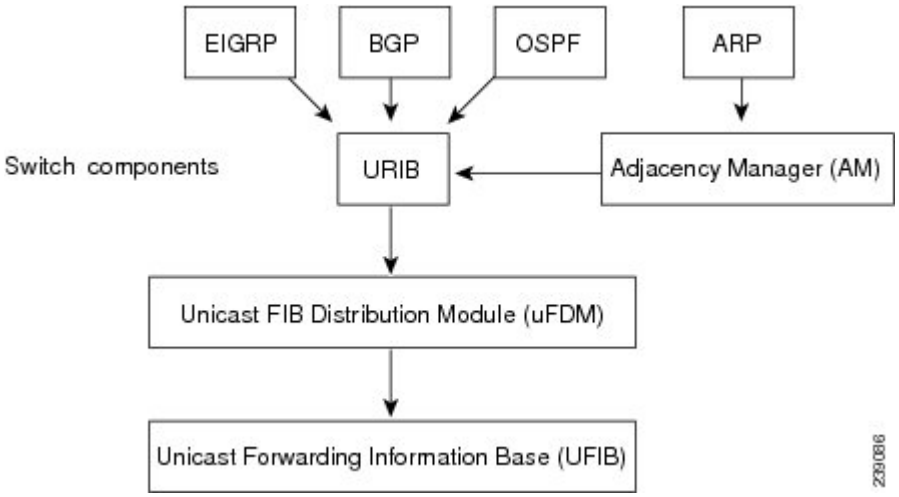
Cisco NX-OS Fowarding Architecture

The Cisco NX-OS forwarding architecture is responsible for processing all routing updates and populating the forwarding information on the switch.

Unicast RIB

The Cisco NX-OS forwarding architecture consists of multiple components, as shown in the following figure.

Figure 3: Cisco NX-OS Forwarding Architecture



The unicast RIB maintains the routing table with directly connected routes, static routes, and routes learned from dynamic unicast routing protocols. The unicast RIB also collects adjacency information from sources such as the Address Resolution Protocol (ARP). The unicast RIB determines the best next-hop for a given route and populates the unicast forwarding information base (FIB) by using the services of the unicast FIB distribution module (FDM).

Each dynamic routing protocol must update the unicast RIB for any route that has timed out. The unicast RIB then deletes that route and recalculates the best next-hop for that route (if an alternate path is available).

Adjacency Manager

The adjacency manager maintains adjacency information for different protocols including ARP, Neighbor Discovery Protocol (NDP), and static configurations. The most basic adjacency information is the Layer 3 to Layer 2 address mapping discovered by these protocols. Outgoing Layer 2 packets use the adjacency information to complete the Layer 2 header.

The ARP process triggers ARP requests to find a particular Layer 3 to Layer 2 mapping. The new mapping becomes available in the adjacency manager when the corresponding ARP reply is received and processed.

Unicast Forwarding Distribution Module

The unicast forwarding distribution module distributes the forwarding path information from the unicast RIB and other sources. The unicast RIB generates forwarding information which the unicast FIB programs into the hardware forwarding tables. The unicast forwarding distribution module also downloads the FIB information to newly inserted modules.

The unicast forwarding distribution module gathers adjacency information, rewrite information, and other platform-dependent information when updating routes in the unicast FIB. The adjacency and rewrite information consists of interface, next-hop, and Layer 3 to Layer 2 mapping information. The interface and next-hop information is received in route updates from the unicast RIB. The Layer 3 to Layer 2 mapping is received from the adjacency manager.

FIB

The unicast FIB builds the information used for the hardware forwarding engine. The unicast FIB receives route updates from the unicast forwarding distribution module and sends the information along to be programmed in the hardware forwarding engine. The unicast FIB controls the addition, deletion, and modification of routes, paths, and adjacencies.

The unicast FIBs are maintained on a per-VRF and per-address-family basis. Based on route update messages, the unicast FIB maintains a per-VRF prefix and next-hop adjacency information database. The next-hop adjacency data structure contains the next-hop IP address and the Layer 2 rewrite information. Multiple prefixes could share a next-hop adjacency information structure.

The unicast FIB also enables and disables unicast reverse path forwarding (RPF) checks per interface. The Cisco Nexus 3600 platform switches support the following two RPF modes that can be configured on each ingress interface:

- RPF Strict Check—Packets that do not have a verifiable source address in the routers forwarding table or do not arrive on any of the return paths to the source are dropped.
- RPF Loose Check—Packets have a verifiable source address in the routers forwarding table and the source is reachable through a physical interface. The ingress interface that receives the packet does not need to match any of the interfaces in the FIB.

Hardware Forwarding

Cisco NX-OS supports distributed packet forwarding. The ingress port takes relevant information from the packet header and passes the information to the local switching engine. The local switching engine does the Layer 3 lookup and uses this information to rewrite the packet header. The ingress module forwards the packet to the egress port. If the egress port is on a different module, the packet is forwarded using the switch fabric to the egress module. The egress module does not participate in the Layer 3 forwarding decision.

Software Forwarding

The software forwarding path in Cisco NX-OS is used mainly to handle features that are not supported in the hardware or to handle errors encountered during the hardware processing. Typically, packets with IP options or packets that need fragmentation are passed to the CPU. The unicast RIB and the adjacency manager make the forwarding decisions based on the packets that should be switched in software or terminated.

Software forwarding is controlled by control plane policies and rate limiters.

Summary of Layer 3 Unicast Routing Features

This section provides a brief introduction to the Layer 3 unicast features and protocols supported in Cisco NX-OS.

Layer 3 uses the IPv4 protocol. For more information, see Configuring IPv4.

IPv4 and IPv6

Layer 3 uses either the IPv4 or IPv6 protocol. IPv6 increases the number of network address bits from 32 bits (in IPv4) to 128 bits. For more information, see Configuring IPv4, on page 17 or Configuring IPv6, on page 35.

OSPF

The OSPF protocol is a link-state routing protocol used to exchange network reachability information within an autonomous system. Each OSPF router advertises information about its active links to its neighbor routers. Link information consists of the link type, the link metric, and the neighbor router connected to the link. The advertisements that contain this link information are called link-state advertisements. For more information, see Configuring OSPFv2 or Configuring OSPFv3, on page 97.

EIGRP

The EIGRP protocol is a unicast routing protocol that has the characteristics of both distance vector and link-state routing protocols. It is an improved version of IGRP, which is a Cisco proprietary routing protocol. EIGRP relies on its neighbors to provide the routes, typical to a distance vector routing protocol. It constructs the network topology from the routes advertised by its neighbors, similar to a link-state protocol, and uses this information to select loop-free paths to destinations. For more information, see Configuring EIGRP.

BGP

The Border Gateway Protocol (BGP) is an inter-autonomous system routing protocol. A BGP router advertises network reachability information to other BGP routers using the Transmission Control Protocol (TCP) as its reliable transport mechanism. The network reachability information includes the destination network prefix, a list of autonomous systems that needs to be traversed to reach the destination, and the next-hop router. Reachability information contains additional path attributes such as the preference to a route, origin of the route, community and others. For more information, see Configuring Basic BGP and Configuring Advanced BGP.

Static Routing

Static routing allows you to enter a fixed route to a destination. This feature is useful for small networks where the topology is simple. Static routing is also used with other routing protocols to control default routes and route distribution. For more information, see Configuring Static Routing.

Layer 3 Virtualization

Virtualization allows you to share physical resources across separate management domains.

Cisco NX-OS supports Layer 3 virtualization with VPN Routing and Forwarding (VRF). A VRF provides a separate address domain for configuring Layer 3 routing protocols. For more information, see Configuring Layer 3 Virtualization.

Route Policy Manager

The Route Policy Manager provides a route filtering capability in Cisco NX-OS. It uses route maps to filter routes distributed across various routing protocols and between different entities within a given routing protocol. Filtering is based on specific match criteria, which is similar to packet filtering by access control lists. For more information, see Configuring Route Policy Manager.

First-Hop Redundancy Protocols

A first-hop redundancy protocol (FHRP) allows you to provide redundant connections to your hosts. If an active first-hop router fails, the FHRP automatically selects a standby router to take over. You do not need to update the hosts with new IP addresses because the address is virtual and shared between each router in the FHRP group. For more information on the Hot Standby Router Protocol (HSRP), see Configuring HSRP For more information on the Virtual Router Redundancy Protocol (VRRP), see Configuring VRRP.

Object Tracking

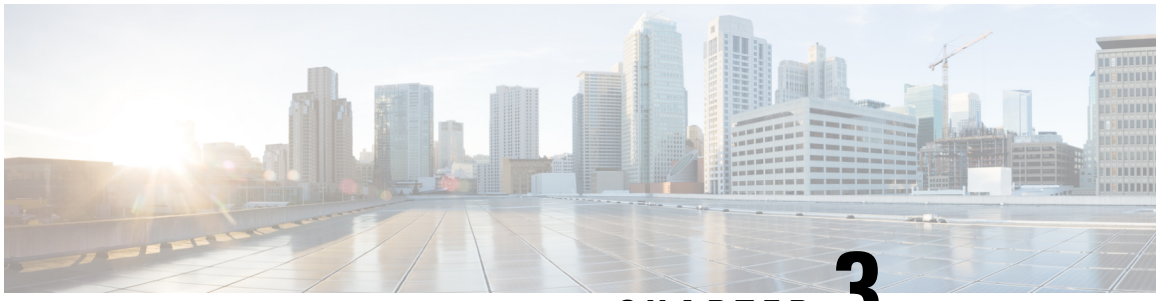
Object tracking allows you to track specific objects on the network, such as the interface line protocol state, IP routing, and route reachability, and take action when the tracked object's state changes. This feature allows you to increase the availability of the network and shorten recovery time if an object state goes down. For more information, see Configuring Object Tracking.

Related Topics

The following Cisco documents are related to the Layer 3 features:

- Exploring Autonomous System Numbers

Related Topics



CHAPTER 3

Configuring IPv4

This chapter describes how to configure Internet Protocol version 4 (IPv4), which includes addressing, Address Resolution Protocol (ARP), and Internet Control Message Protocol (ICMP), on the Cisco NX-OS switch.

This chapter includes the following sections:

- About IPv4, on page 17
- Prerequisites for IPv4, on page 22
- Guidelines and Limitations, on page 22
- Default Settings, on page 22
- Configuring IPv4, on page 23
- Verifying the IPv4 Configuration, on page 32
- Configuration Examples for IPv4, on page 33

About IPv4

You can configure IP on the switch to assign IP addresses to network interfaces. When you assign IP addresses, you enable the interfaces and allow communication with the hosts on those interfaces.

You can configure an IP address as primary or secondary on a switch. An interface can have one primary IP address and multiple secondary addresses. All networking switches on an interface should share the same primary IP address because the packets that are generated by the switch always use the primary IPv4 address. Each IPv4 packet is based on the information from a source or destination IP address. See the Multiple IPv4 Addresses section.

You can use a subnet to mask the IP addresses. A mask is used to determine what subnet an IP address belongs to. An IP address contains the network address and the host address. A mask identifies the bits that denote the network number in an IP address. When you use the mask to subnet a network, the mask is then referred to as a subnet mask. Subnet masks are 32-bit values that allow the recipient of IP packets to distinguish the network ID portion of the IP address from the host ID portion of the IP address.

The IP feature in the Cisco NX-OS system is responsible for handling IPv4 packets, as well as the forwarding of IPv4 packets, which includes IPv4 unicast and multicast route lookup, reverse path forwarding (RPF) checks, and software access control list (ACL) forwarding. The IP feature also manages the network interface IP address configuration, duplicate address checks, static routes, and packet send and receive interface for IP clients.

Multiple IPv4 Addresses

The Cisco NX-OS system supports multiple IP addresses per interface. You can specify an unlimited number of secondary addresses for a variety of situations. The most common situations are as follows:

- When there are not enough host IP addresses for a particular network interface. For example, if your subnet allows up to 254 hosts per logical subnet, but on one physical subnet you must have 300 host addresses, then you can use secondary IP addresses on the routers or access servers to allow you to have two logical subnets using one physical subnet.
- Two subnets of a single network might otherwise be separated by another network. You can create a single network from subnets that are physically separated by another network by using a secondary address. In these instances, the first network is extended, or layered on top of the second network. A subnet cannot appear on more than one active interface of the router at a time.

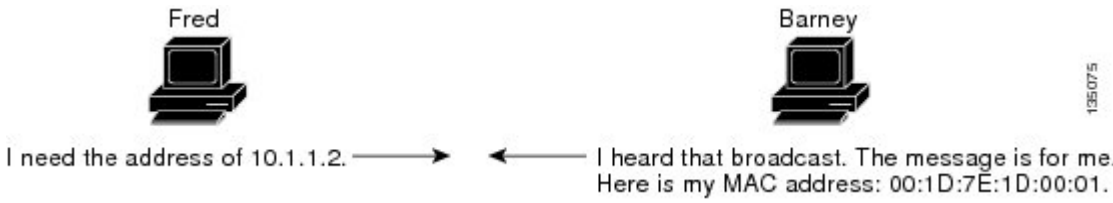
Note If any switch on a network segment uses a secondary IPv4 address, all other switches on that same network interface must also use a secondary address from the same network or subnet. The inconsistent use of secondary addresses on a network segment can quickly cause routing loops.

Address Resolution Protocol

Networking switches and Layer 3 switches use Address Resolution Protocol (ARP) to map IP (network layer) addresses to (Media Access Control [MAC]-layer) addresses to enable IP packets to be sent across networks. Before a switch sends a packet to another switch, it looks in its own ARP cache to see if there is a MAC address and corresponding IP address for the destination switch. If there is no entry, the source switch sends a broadcast message to every switch on the network.

Each switch compares the IP address to its own. Only the switch with the matching IP address replies to the switch that sends the data with a packet that contains the MAC address for the switch. The source switch adds the destination switch MAC address to its ARP table for future reference, creates a data-link header and trailer that encapsulates the packet, and proceeds to transfer the data. The following figure shows the ARP broadcast and response process.

Figure 4: ARP Process



When the destination switch lies on a remote network which is beyond another router, the process is the same except that the switch that sends the data sends an ARP request for the MAC address of the default gateway. After the address is resolved and the default gateway receives the data packet, the default gateway broadcasts the destination IP address over the networks connected to it. The switch on the destination switch network uses ARP to obtain the MAC address of the destination switch and delivers the packet. ARP is enabled by default.

The default system-defined CoPP policy rate-limits ARP broadcast packets. The default system-defined CoPP policy prevents an ARP broadcast storm from affecting the control plane traffic but does not affect bridged packets.

ARP Caching

ARP caching minimizes broadcasts and limits wasteful use of network resources. The mapping of IP addresses to MAC addresses occurs at each hop (switch) on the network for every packet sent over an internetwork, which may affect network performance.

ARP caching stores network addresses and the associated data-link addresses in memory for a period of time, which minimizes the use of valuable network resources to broadcast for the same address each time a packet is sent. You must maintain the cache entries since the cache entries are set to expire periodically because the information might become outdated. Every switch on a network updates its tables as addresses are broadcast.

Devices That Do Not Use ARP

When a network is divided into two segments, a bridge joins the segments and filters traffic to each segment based on MAC addresses. The bridge builds its own address table that uses MAC addresses only, as opposed to a switch, which has an ARP cache that contains both IP addresses and the corresponding MAC addresses.

Passive hubs are central-connection switches that physically connect other switches in a network. They send messages out on all their ports to the switches and operate at Layer 1 but do not maintain an address table.

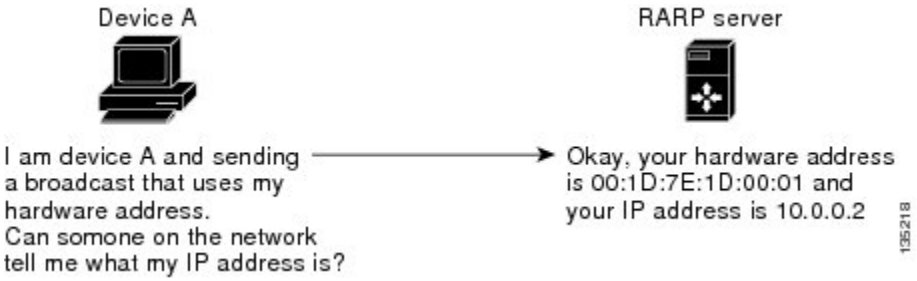
Layer 2 switches determine which port is connected to a device to which the message is addressed and send only to that port, unlike a hub, which sends the message out all of its ports. However, Layer 3 switches are switches that build an ARP cache (table).

Reverse ARP

Reverse ARP (RARP) as defined by RFC 903 works the same way as ARP, except that the RARP request packet requests an IP address instead of a MAC address. RARP often is used by diskless workstations because this type of device has no way to store IP addresses to use when they boot. The only address that is known is the MAC address because it is burned into the hardware.

Use of RARP requires an RARP server on the same network segment as the router interface. The following figure illustrates how RARP works.

Figure 5: Reverse ARP



There are several limitations of RARP. Because of these limitations, most businesses use DHCP to assign IP addresses dynamically. DHCP is cost effective and requires less maintenance than RARP. The following are the most important limitations:

- Because RARP uses hardware addresses, if the internetwork is large with many physical networks, a RARP server must be on every segment with an additional server for redundancy. Maintaining two servers for every segment is costly.
- Each server must be configured with a table of static mappings between the hardware addresses and IP addresses. Maintenance of the IP addresses is difficult.
- RARP only provides IP addresses of the hosts and not subnet masks or default gateways.

Proxy ARP

Proxy ARP enables a switch that is physically located on one network appear to be logically part of a different physical network connected to the same switch or firewall. Proxy ARP allows you to hide a switch with a public IP address on a private network behind a router and still have the switch appear to be on the public network in front of the router. By hiding its identity, the router accepts responsibility for routing packets to the real destination. Proxy ARP can help switches on a subnet reach remote subnets without configuring routing or a default gateway.

When switches are not in the same data link layer network but in the same IP network, they try to transmit data to each other as if they are on the local network. However, the router that separates the switches does not send a broadcast message because routers do not pass hardware-layer broadcasts and the addresses cannot be resolved.

When you enable Proxy ARP on the switch and it receives an ARP request, it identifies the request as a request for a system that is not on the local LAN. The switch responds as if it is the remote destination for which the broadcast is addressed, with an ARP response that associates the MAC address of the switch with the IP address of the remote destination. The local switch believes that it is directly connected to the destination, while in reality its packets are being forwarded from the local subnetwork toward the destination subnetwork by their local switch. By default, Proxy ARP is disabled.

Local Proxy ARP

You can use local Proxy ARP to enable a switch to respond to ARP requests for IP addresses within a subnet where normally no routing is required. When you enable local Proxy ARP, ARP responds to all ARP requests for IP addresses within the subnet and forwards all traffic between hosts in the subnet. Use this feature only on subnets where hosts are intentionally prevented from communicating directly by the configuration on the switch to which they are connected.

Gratuitous ARP

Gratuitous ARP sends a request with identical source IP address and destination IP address to detect duplicate IP addresses. Cisco NX-OS supports enabling or disabling gratuitous ARP requests or ARP cache updates.

Glean Throttling

When forwarding an incoming IP packet, if the Address Resolution Protocol (ARP) request for the next-hop is not resolved, packets are punted to the central processing unit (CPU) for ARP resolution. The CPU resolves the MAC address for the next-hop and programs the hardware.

The device hardware has glean rate limiters to protect the supervisor from the glean traffic. If the maximum number of entries is exceeded, the packets for which the ARP request is not resolved continues to be processed in the software instead of getting dropped in the hardware.

When an ARP request is sent, the software adds a /32 drop adjacency in the hardware to prevent the packets to the same next-hop IP address to be forwarded to the supervisor. When the ARP is resolved, the hardware entry is updated with the correct MAC address. If the ARP entry is not resolved before a timeout period, the entry is removed from the hardware.

Note Glean throttling is supported for IPv4 and IPv6, but IPv6 link-local addresses are not supported.

ICMP

You can use ICMP to provide message packets that report errors and other information that is relevant to IP processing. ICMP generates error messages, such as ICMP destination unreachable messages, ICMP Echo Requests (which send a packet on a round trip between two hosts) and Echo Reply messages. ICMP also provides many diagnostic functions and can send and redirect error packets to the host. By default, ICMP is enabled.

Some of the ICMP message types are as follows:

- Network error messages
- Network congestion messages
- Troubleshooting information
- Timeout announcements

Note ICMP redirects are disabled on interfaces where the local proxy ARP feature is enabled.

ICMP Unreachable Support to Set Source Interface

You can configure an interface IP address for the ICMP source IP field to handle ICMP error messages. When ICMP packets are constructed in a network stack, the packets use the configured interface IP address. You can select Ethernet, loopback, or port channel interfaces to configure the IP address.

Virtualization Support

IPv4 supports Virtual Routing and Forwarding instances (VRFs). By default, Cisco NX-OS places you in the default VRF unless you specifically configure another VRF. For more information, see Configuring Layer 3 Virtualization.

IPv4 Routes with ECMP

If all next-hops for a route are glean, drop, or punt, all next-hops are programmed as-is in the Multipath hardware table.

If some next-hops for a route are glean, drop, or punt, and the remaining next-hops are not, then only non glean, drop, or punt next-hops are programmed in the Multipath hardware table.

When a specific next-hop for ECMP route is resolved (ARP ND resolved), then the Multipath hardware table is updated accordingly.

Prerequisites for IPv4

IPv4 has the following prerequisites:

- IPv4 can only be configured on Layer 3 interfaces.

Guidelines and Limitations

IPv4 has the following configuration guidelines and limitations:

- You can configure a secondary IP address only after you configure the primary IP address.
- If the switch is used as a Layer 2 or Layer 3 termination switch, Cisco recommends that you set the **mac-address-table-aging-time** to 1800 (higher than the default ARP aging time of 1500 seconds) on all VLANs.
- The switch does not support per-VLAN cam aging timers.

Default Settings

The following table lists the default settings for IP parameters.

Table 3: Default IP Parameters

Parameters	Default
ARP timeout	1500 seconds
Proxy ARP	Disabled

Configuring IPv4

Configuring IPv4 Addressing

You can assign a primary IP address for a network interface.

Procedure

	Command or Action	Purpose

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

Step 2	**interface ethernet number**
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 4	**ip address** *ip-address* / *length* [**secondary**] **Example:** `switch(config-if)# ip address 192.2.1.1 255.0.0.0`	Specifies a primary or secondary IPv4 address for an interface. • The network mask can be a four-part dotted decimal address. For example, 255.0.0.0 indicates that each bit equal to 1 means the corresponding address bit belongs to the network address. • The network mask can be indicated as a slash (/) and a number - a prefix length. The prefix length is a decimal value that indicates how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address). A slash must precede the decimal value and there is no space between the IP address and the slash.
Step 5	(Optional) **show ip interface** **Example:** `switch(config-if)# show ip interface`	Displays interfaces configured for IPv4.

	Command or Action	Purpose
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to assign an IPv4 address:

```
switch# configure terminal
switch(config)# interface ethernet 2/3
switch(config-if)# no switchport
switch(config-if)# ip address 192.2.1.1 255.0.0.0
switch(config-if)# copy running-config startup-config
```

Configuring Multiple IP Addresses

You can only add secondary IP addresses after you configure primary IP addresses.

Procedure

	Command or Action	Purpose

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

Step 2	**interface ethernet number**
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 4	**ip address** *ip-address / length* [**secondary**] **Example:** `switch(config-if)# ip address 192.2.1.1 255.0.0.0 secondary`	Specifies the configured address as a secondary IPv4 address.
Step 5	(Optional) **show ip interface** **Example:** `switch(config-if)# show ip interface`	Displays interfaces configured for IPv4.

	Command or Action	Purpose
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Configuring a Static ARP Entry

You can configure a static ARP entry on the switch to map IP addresses to MAC hardware addresses, including static multicast MAC addresses.

Procedure

	Command or Action	Purpose

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

Step 2	**interface ethernet number**
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 4	**ip arp ipaddr mac_addr** **Example:** `switch(config-if)# ip arp 192.2.1.1 0019.076c.1a78`	Associates an IP address with a MAC address as a static entry
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure a static ARP entry:

```
switch# configure terminal
switch(config)# interface ethernet 2/3
switch(config-if)# no switchport
switch(config-if)# ip arp 1 92.2.1.1 0019.076c.1a78
switch(config-if)# copy running-config startup-config
```

Configuring Proxy ARP

You can configure Proxy ARP on the switch to determine the media addresses of hosts on other networks or subnets.

Procedure

	Command or Action	Purpose

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

Step 2	**interface ethernet number**
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 4	**ip proxy-arp** **Example:** `switch(config-if)# ip proxy-arp`	Enables Proxy ARP on the interface.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure Proxy ARP:

```
switch# configure terminal
switch(config)# interface ethernet 2/3
switch(config-if)# no switchport
switch(config-if)# ip proxy-arp
switch(config-if)# copy running-config startup-config
```

Configuring Local Proxy ARP

You can configure Local Proxy ARP on the switch.

Procedure

	Command or Action	Purpose

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

Step 2	**interface ethernet number**
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface
Step 4	**ip local-proxy-arp** **Example:** `switch(config-if)# ip local-proxy-arp`	Enables Local Proxy ARP on the interface.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure Local Proxy ARP:

```
switch# configure terminal
switch(config)# interface ethernet 2/3
switch(config-if)# no switchport
switch(config-if)# ip local-proxy-arp
switch(config-if)# copy running-config startup-config
```

Configuring Gratuitous ARP

You can configure gratuitous ARP on an interface.

Procedure

	Command or Action	Purpose

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

	Command or Action	Purpose

Step 2	**interface ethernet number**
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 4	**ip arp gratuitous** { **request** | **update** } **Example:** `switch(config-if)# ip arp gratuitous request`	Enables gratuitous ARP on the interface. Default is enabled.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to enable IP glean throttling:

```
switch# configure terminal
switch(config)# hardware ip glean throttle
switch(config-if)# copy running-config startup-config
```

Configuring IP Directed Broadcasts

An IP directed broadcast is an IP packet whose destination address is a valid broadcast address for some IP subnet, but which originates from a node that is not itself part of that destination subnet.

A switch that is not directly connected to its destination subnet forwards an IP directed broadcast in the same way it would forward unicast IP packets destined to a host on that subnet. When a directed broadcast packet reaches a switch that is directly connected to its destination subnet, that packet is “exploded” as a broadcast on the destination subnet. The destination address in the IP header of the packet is rewritten to the configured IP broadcast address for the subnet, and the packet is sent as a link-layer broadcast.

If directed broadcast is enabled for an interface, incoming IP packets whose addresses identify them as directed broadcasts intended for the subnet to which that interface is attached will be exploded as broadcasts on that subnet.

To enable IP directed broadcasts, use the following command in interface configuration mode:

Command	Purpose
ip directed-broadcast	Enables the translation of a directed broadcast to physical broadcasts

Configuring IP Glean Throttling

Cisco NX-OS software supports glean throttling rate limiters to protect the supervisor from the glean traffic.

Note We recommend that you configure the IP glean throttle feature by using the hardware ip glean throttle command to filter the unnecessary glean packets that are sent to the supervisor for ARP resolution for the next-hops that are not reachable or do not exist. IP glean throttling boosts software performance and helps to manage traffic more efficiently.

Note Glean throttling is supported for IPv4 and IPv6, but IPv6 link-local addresses are not supported.

Procedure

	Command or Action	Purpose

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

Step 2	**hardware ip glean throttle** **Example:** `switch(config)# hardware ip glean throttle`	Enables ARP throttling.
Step 3	**no hardware ip glean throttle** **Example:** `switch(config)# no hardware ip glean throttle`	Disables ARP throttling.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to enable IP glean throttling:

```
switch# configure terminal
switch(config)# hardware ip glean throttle
switch(config-if)# copy running-config startup-config
```

Configuring the Hardware IP Glean Throttle Timeout

You can configure a timeout for the installed drop adjacencies to remain in the FIB.

Procedure

	Command or Action	Purpose

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

Step 2	**hardware ip glean throttle maximum timeout** *timeout-in-seconds* **Example:** `switch(config)# hardware ip glean throttle maximum timeout 300`	Configures the timeout for the installed drop adjacencies to remain in the FIB.
Step 3	[**no**] **hardware ip glean throttle maximum timeout** *timeout-in-seconds* **Example:** `switch(config)# no hardware ip glean throttle maximum timeout 300`	Applies the default limits. The timeout value is in seconds. The range is from 300 seconds (5 minutes) to 1800 seconds (30 minutes). **Note** After the timeout period is exceeded, the drop adjacencies are removed from the FIB.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to disable gratuitous ARP requests:

```
switch# configure terminal
switch(config)# hardware ip glean throttle maximum timeout 300
switch(config-if)# copy running-config startup-config
```

Configuring the Interface IP Address for the ICMP Source IP Field

You can configure an interface IP address for the ICMP source IP field to handle ICMP error messages.

Procedure

	Command or Action	Purpose

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

Step 2	[**no**] **ip source** {**ethernet** *slot/port* | **loopback** *number* | **port-channel** *number*} {**icmp-errors**} **Example:** `switch(config)# ip source loopback 0 icmp-errors`	Configures an interface IP address for the ICMP source IP field to route ICMP error messages.

Example

This example shows how to configure an interface IP address for the ICMP source IP field:

```
switch# configure terminal
switch(config)# ip source ethernet 1/1 icmp-errors
```

This example shows how to configure an interface IP address for the ICMP source IP field:

```
switch# configure terminal
switch(config)# no ip source ethernet 1/1 icmp-errors
```

Configuring Logging for Software Forwarding of IP Packets

You can configure the logging conditions for IP packets that are forwarded by the NX-OS software. The conditions consist of the following:

- A minimum number of packets (the size)
- An optional period of time (the logging interval)

The logging conditions create the packet per second (pps) threshold. When traffic meets or exceeds the conditions, NX-OS logs a console message. For example:

```
2019 jul 31 15:28:31 switch-1 %$ VDC-1 %$ %USER-3-SYSTEM_MSG: Packets per second exceeded
the configured threshold 40, current PPS: 1262  - netstack
```

You can set the conditions for forwarded packets through the **ip pps threshold unicast-forward** command. To disable the feature, use **no ip pps threshold unicast-forward**.

Procedure

	Command or Action	Purpose
Step 1	**config terminal** **Example:**	Enters the configuration terminal.

	Command or Action	Purpose

	`switch-1# config terminal`
	`Enter configuration commands, one per line. End with CNTL/Z.`
	`switch-1(config)#`	

Step 2	**ip pps threshold unicast-forward** *pps-threshold* [*syslog-interval*]
	Example:
	`switch-1(config)# ip pps threshold unicast-forward 50 5`
	`switch-1(config)#`
		Enable the feature and set the conditions:
		• The *pps-threshold* is from 1 through 30000 packets.
		• The *syslog-interval* is from 1 through 60 seconds. The default is 1 second.

Step 3	(Optional) **show ip pps threshold**
	Example:
	`switch-1(config) show ip traffic pps`
	`PPS type : unicast-forward, PPS limit : 50, Log Interval: 5`
	`switch-1(config)#`
		Display the current PPS threshold configuration.

Example

This example shows how to configure a console message if the number of packets that get forwarded for a specific flow exceeds the configured packet count and a logging interval of 4000 packets every 2 seconds:

```
switch-1# configure terminal
switch-1(config)# ip pps threshold unicast-forward 4000 2
switch-1(config)# copy running-config startup-config
```

Verifying the IPv4 Configuration

To display the IPv4 configuration information, perform one of the following tasks:

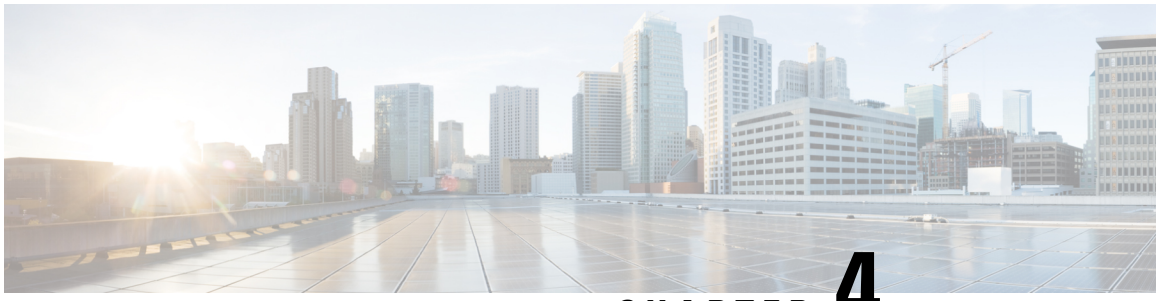
Command	Purpose
how hardware forwarding ip verify	Displays the IP packet verification configuration.
show ip adjacency	Displays the adjacency table.
show ip arp	Displays the ARP table.
show ip interface	Displays IP-related interface information.
show ip arp statistics [**vrf** *vrf-name*]	Displays the ARP statistics.
show ip adjacency summary	Displays the summary of number of throttle adjacencies.
show ip arp summary	Displays the summary of the number of throttle adjacencies.

Command	Purpose
show ip interface	Displays IP-related interface information.

Configuration Examples for IPv4

This example shows how to configure an IPv4 address:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config)# no switchport
switch(config-if)#ip address 192.2.1.1/1
```



CHAPTER 4

Configuring IPv6

This chapter describes how to configure Internet Protocol version 6 (IPv6), which includes addressing, Neighbor Discovery Protocol (ND), and Internet Control Message Protocol version 6 (ICMPv6), on the Cisco NX-OS device.

This chapter includes the following sections:

- About IPv6, on page 35
- Prerequisites for IPv6, on page 49
- Guidelines and Limitations for IPv6, on page 49
- Default Settings, on page 49
- Configuring IPv6, on page 50
- Verifying the IPv6 Configuration, on page 56
- Configuration Examples for IPv6, on page 56

About IPv6

IPv6, which is designed to replace IPv4, increases the number of network address bits from 32 bits (in IPv4) to 128 bits. IPv6 is based on IPv4 but it includes a much larger address space and other improvements such as a simplified main header and extension headers.

The larger IPv6 address space allows networks to scale and provide global reachability. The simplified IPv6 packet header format handles packets more efficiently. The flexibility of the IPv6 address space reduces the need for private addresses and the use of Network Address Translation (NAT), which translates private (not globally unique) addresses into a limited number of public addresses. IPv6 enables new application protocols that do not require special processing by border routers at the edge of networks.

IPv6 functionality, such as prefix aggregation, simplified network renumbering, and IPv6 site multihoming capabilities, enable more efficient routing. IPv6 supports Open Shortest Path First (OSPF) for IPv6 and multiprotocol Border Gateway Protocol (BGP).

IPv6 Address Formats

An IPv6 address has 128 bits or 16 bytes. The address is divided into eight, 16-bit hexadecimal blocks separated by colons (:) in the format: x:x:x:x:x:x:x:x. Two examples of IPv6 addresses are as follows:

```
2001:0DB8:7654:3210:FEDC:BA98:7654:32102001:0DB8:0:0:8:800:200C:417A
```

IPv6 addresses contain consecutive zeros within the address. You can use two colons (::) at the beginning, middle, or end of an IPv6 address to replace the consecutive zeros. The following table shows a list of compressed IPv6 address formats.

Note You can use two colons (::) only once in an IPv6 address to replace the longest string of consecutive zeros within the address.

You can use a double colon as part of the IPv6 address when consecutive 16-bit values are denoted as zero. You can configure multiple IPv6 addresses per interface but only one link-local address.

The hexadecimal letters in IPv6 addresses are not case sensitive.

Table 4: Compressed IPv6 Address Formats

IPv6 Address Type	Preferred Format	Compressed Format
Unicast	2001:0:0:0:0DB8:800:200C:417A	2001::0DB8:800:200C:417A
Multicast	FF01:0:0:0:0:0:0:101	FF01::101
Loopback	0:0:0:0:0:0:0:1	::1
Unspecified	0:0:0:0:0:0:0:0	::

A node may use the loopback address listed in the table **Compressed IPv6 Address Formats** to send an IPv6 packet to itself. The loopback address in IPv6 is the same as the loopback address in IPv4. For more information, see Overview, on page 3.

Note You cannot assign the IPv6 loopback address to a physical interface. A packet that contains the IPv6 loopback address as its source or destination address must remain within the node that created the packet. IPv6 routers do not forward packets that have the IPv6 loopback address as their source or destination address.

Note You cannot assign an IPv6 unspecified address to an interface. You should not use the unspecified IPv6 addresses as destination addresses in IPv6 packets or the IPv6 routing header.

The IPv6-prefix is in the form documented in RFC 2373 where the IPv6 address is specified in hexadecimal using 16-bit values between colons. The prefix length is a decimal value that indicates how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address). For example, 2001:0DB8:8086:6502::/32 is a valid IPv6 prefix.

IPv6 Unicast Addresses

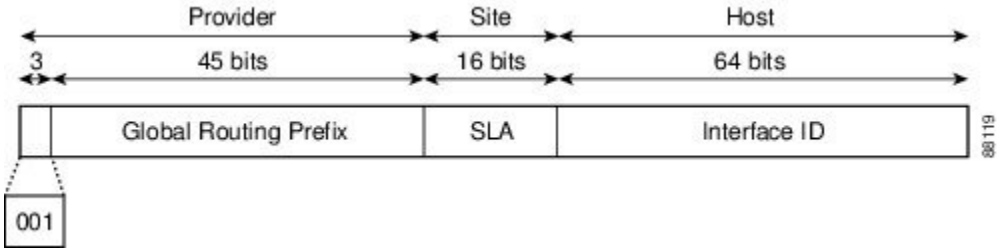
An IPv6 unicast address is an identifier for a single interface on a single node. A packet that is sent to a unicast address is delivered to the interface identified by that address.

Aggregatable Global Addresses

An aggregatable global address is an IPv6 address from the aggregatable global unicast prefix. The structure of aggregatable global unicast addresses enables strict aggregation of routing prefixes that limits the number of routing table entries in the global routing table. Aggregatable global addresses are used on links that are aggregated upward through organizations and eventually to the Internet service providers (ISPs).

Aggregatable global IPv6 addresses are defined by a global routing prefix, a subnet ID, and an interface ID. Except for addresses that start with binary 000, all global unicast addresses have a 64-bit interface ID. The IPv6 global unicast address allocation uses the range of addresses that start with binary value 001 (2000::/3). The following figure shows the structure of an aggregatable global address.

Figure 6: Aggregatable Global Address Format



Addresses with a prefix of 2000::/3 (001) through E000::/3 (111) are required to have 64-bit interface identifiers in the extended universal identifier (EUI)-64 format. The Internet Assigned Numbers Authority (IANA) allocates the IPv6 address space in the range of 2000::/16 to regional registries.

The aggregatable global address consists of a 48-bit global routing prefix and a 16-bit subnet ID or Site-Level Aggregator (SLA). In the IPv6 aggregatable global unicast address format document (RFC 2374), the global routing prefix included two other hierarchically structured fields called Top-Level Aggregator (TLA) and Next-Level Aggregator (NLA). The IETF decided to remove the TLS and NLA fields from the RFCs because these fields are policy based. Some existing IPv6 networks deployed before the change might still use networks that are on the older architecture.

A subnet ID, which is a 16-bit subnet field, can be used by individual organizations to create a local addressing hierarchy and to identify subnets. A subnet ID is similar to a subnet in IPv4, except that an organization with an IPv6 subnet ID can support up to 65,535 individual subnets.

An interface ID identifies interfaces on a link. The interface ID is unique to the link. In many cases, an interface ID is the same as or based on the link-layer address of an interface. Interface IDs used in aggregatable global unicast and other IPv6 address types have 64 bits and are in the modified EUI-64 format.

Interface IDs are in the modified EUI-64 format in one of the following ways:

- For all IEEE 802 interface types (for example, Ethernet, and Fiber Distributed Data interfaces), the first three octets (24 bits) are the Organizationally Unique Identifier (OUI) of the 48-bit link-layer address (MAC address) of the interface, the fourth and fifth octets (16 bits) are a fixed hexadecimal value of FFFE, and the last three octets (24 bits) are the last three octets of the MAC address. The Universal/Local (U/L) bit, which is the seventh bit of the first octet, has a value of 0 or 1. Zero indicates a locally administered identifier; 1 indicates a globally unique IPv6 interface identifier.
- For all other interface types (for example, serial, loopback, ATM, Frame Relay, and tunnel interface types—except tunnel interfaces used with IPv6 overlay tunnels), the interface ID is similar to the interface ID for IEEE 802 interface types; however, the first MAC address from the pool of MAC addresses in the router is used as the identifier (because the interface does not have a MAC address).

- For tunnel interface types that are used with IPv6 overlay tunnels, the interface ID is the IPv4 address assigned to the tunnel interface with all zeros in the high-order 32 bits of the identifier.

Note For interfaces that use the Point-to-Point Protocol (PPP), where the interfaces at both ends of the connection might have the same MAC address, the interface identifiers at both ends of the connection are negotiated (picked randomly and, if necessary, reconstructed) until both identifiers are unique. The first MAC address in the router is used as the identifier for interfaces using PPP.

If no IEEE 802 interface types are in the router, link-local IPv6 addresses are generated on the interfaces in the router in the following sequence:

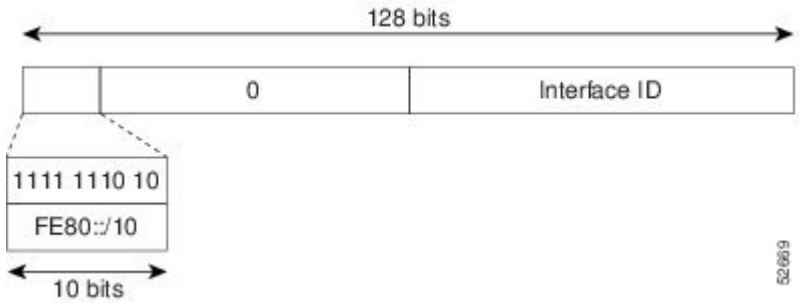
1. The router is queried for MAC addresses (from the pool of MAC addresses in the router).
2. If no MAC addresses are available in the router, the serial number of the router is used to form the link-local addresses.
3. If the serial number of the router cannot be used to form the link-local addresses, the router uses a Message Digest 5 (MD5) hash to determine the MAC address of the router from the hostname of the router.

Link-Local Addresses

A link-local address is an IPv6 unicast address that can be automatically configured on any interface using the link-local prefix FE80::/10 (1111 1110 10) and the interface identifier in the modified EUI-64 format. Link-local addresses are used in the Neighbor Discovery Protocol (NDP). Nodes on a local link can use link-local addresses to communicate; the nodes do not need globally unique addresses to communicate. The following figure shows the structure of a link-local address.

IPv6 routers cannot forward packets that have link-local source or destination addresses to other links.

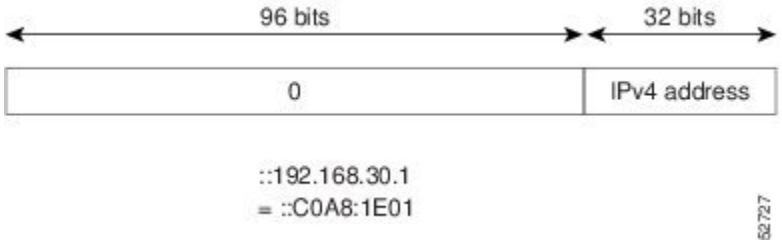
Figure 7: Link-Local Address Format



IPv4-Compatible IPv6 Addresses

An IPv4-compatible IPv6 address is an IPv6 unicast address that has zeros in the high-order 96 bits of the address and an IPv4 address in the low-order 32 bits of the address. The format of an IPv4-compatible IPv6 address is 0:0:0:0:0:0:A.B.C.D or ::A.B.C.D. The entire 128-bit IPv4-compatible IPv6 address is used as the IPv6 address of a node and the IPv4 address embedded in the low-order 32 bits is used as the IPv4 address of the node. IPv4-compatible IPv6 addresses are assigned to nodes that support both the IPv4 and IPv6 protocol stacks and are used in automatic tunnels. The following figure shows the structure of an IPv4-compatible IPv6 address and a few acceptable formats for the address.

Figure 8: IPv4-Compatible IPv6 Address Format



Unique Local Addresses

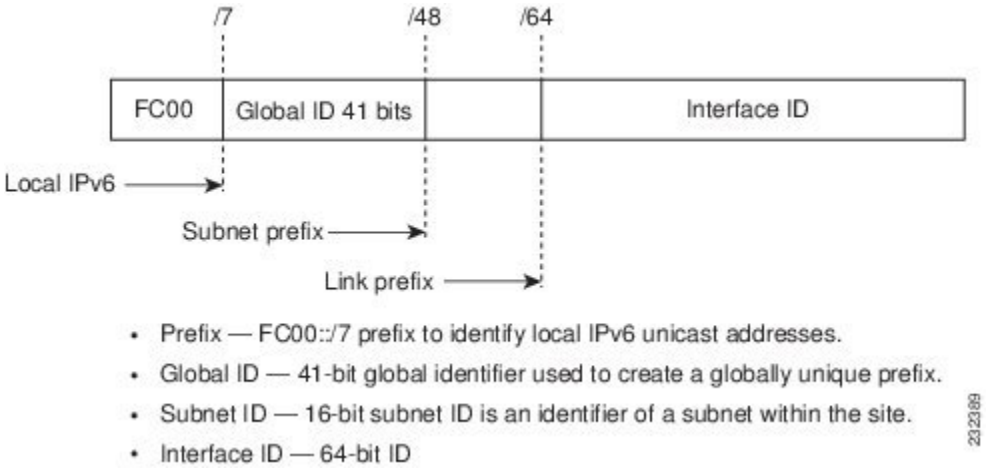
A unique local address is an IPv6 unicast address that is globally unique and is intended for local communications. It is not expected to be routable on the global Internet and is routable inside of a limited area, such as a site, and it may be routed between a limited set of sites. Applications may treat unique local addresses like global scoped addresses.

A unique local address has the following characteristics:

- It has a globally unique prefix (it has a high probability of uniqueness).
- It has a well-known prefix to allow for easy filtering at site boundaries.
- It allows sites to be combined or privately interconnected without creating any address conflicts or requiring renumbering of interfaces that use these prefixes.
- It is ISP-independent and can be used for communications inside of a site without having any permanent or intermittent Internet connectivity.
- If it is accidentally leaked outside of a site through routing or the Domain Name Server (DNS), there is no conflict with any other addresses.

The following figure shows the structure of a unique local address.

Figure 9: Unique Local Address Structure



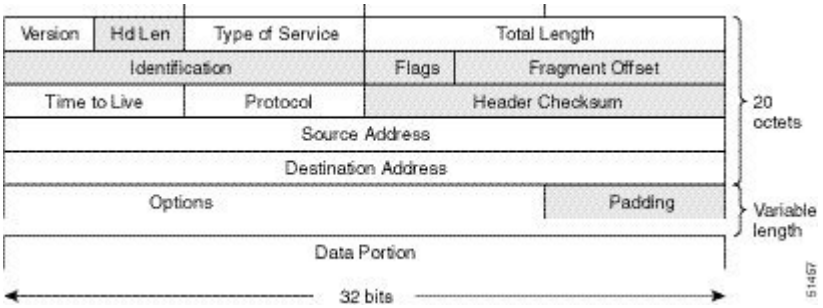
Site-Local Address

Because RFC 3879 deprecates the use of site-local addresses, you should follow the recommendations of unique local addressing (ULA) in RFC 4193 when you configure private IPv6 addresses.

IPv4 Packet Header

The base IPv4 packet header has 12 fields with a total size of 20 octets (160 bits) (see the following figure). The 12 fields may be followed by an Options field, which is followed by a data portion that is usually the transport-layer packet. The variable length of the Options field adds to the total size of the IPv4 packet header. The shaded fields of the IPv4 packet header are not included in the IPv6 packet header.

Figure 10: IPv4 Packet Header Format



Simplified IPv6 Packet Header

The base IPv6 packet header has 8 fields with a total size of 40 octets (320 bits) (see the following figure). Fragmentation is handled by the source of a packet and checksums at the data link layer and transport layer are used. The User Datagram Protocol (UDP) checksum checks the integrity of the inner packet and the base IPv6 packet header and Options field are aligned to 64 bits, which can facilitate the processing of IPv6 packets.

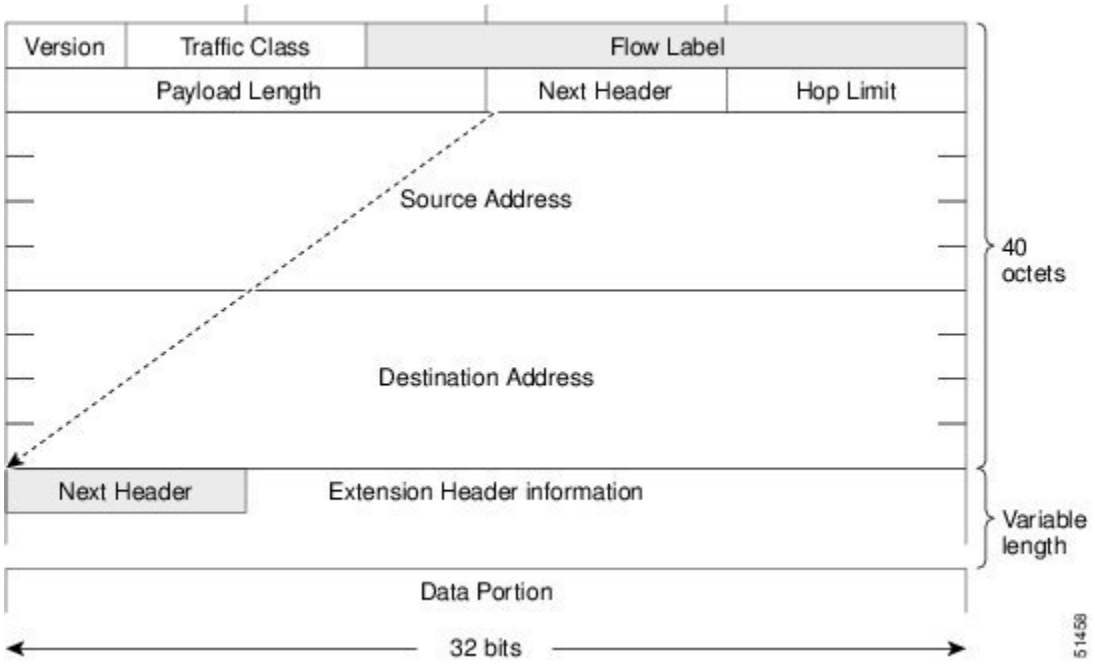
The following table lists the fields in the base IPv6 packet header.

Table 5: Table 3-2 Base IPv6 Packet Header Fields

Field	Description
Version	Similar to the Version field in the IPv4 packet header, except that the field lists number 6 for IPv6 instead of number 4 for IPv4.
Traffic Class	Similar to the Type of Service field in the IPv4 packet header. The Traffic Class field tags packets with a traffic class that is used in differentiated services.
Flow Label	New field in the IPv6 packet header. The Flow Label field tags packets with a specific flow that differentiates the packets at the network layer.
Payload Length	Similar to the Total Length field in the IPv4 packet header. The Payload Length field indicates the total length of the data portion of the packet.

Field	Description
Next Header	Similar to the Protocol field in the IPv4 packet header. The value of the Next Header field determines the type of information that follows the base IPv6 header. The type of information that follows the base IPv6 header can be a transport-layer packet, for example, a TCP or UDP packet, or an Extension Header, as shown in the following figure.
Hop Limit	Similar to the Time to Live field in the IPv4 packet header. The value of the Hop Limit field specifies the maximum number of routers that an IPv6 packet can pass through before the packet is considered invalid. Each router decrements the value by one. Because no checksum is in the IPv6 header, the router can decrement the value without needing to recalculate the checksum, which saves processing resources.
Source Address	Similar to the Source Address field in the IPv4 packet header, except that the field contains a 128-bit source address for IPv6 instead of a 32-bit source address for IPv4.
Destination Address	Similar to the Destination Address field in the IPv4 packet header, except that the field contains a 128-bit destination address for IPv6 instead of a 32-bit destination address for IPv4.

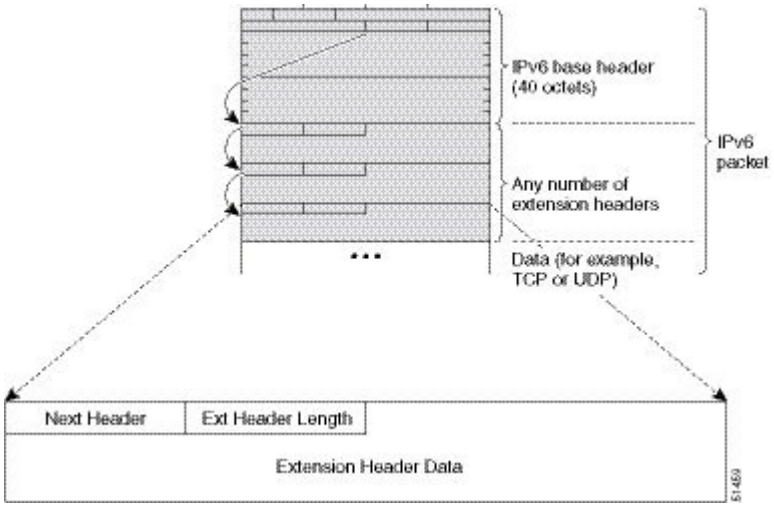
Figure 11: IPv6 Packet Header Format



IPv6 Extension Headers

Optional extension headers and the data portion of the packet are after the eight fields of the base IPv6 packet header. If present, each extension header is aligned to 64 bits. There is no fixed number of extension headers in an IPv6 packet. Each extension header is identified by the Next Header field of the previous header. Typically, the final extension header has a Next Header field of a transport-layer protocol, such as TCP or UDP. The following figure shows the IPv6 extension header format.

Figure 12: IPv6 Extension Header Format



The following table lists the extension header types and their Next Header field values.

Table 6: IPv6 Extension Header Types

Header Type	Next Header Value	Description
Hop-by-hop options header	0	Header that is processed by all hops in the path of a packet. When present, the hop-by-hop options header always follows immediately after the base IPv6 packet header.
Destination options header	60	Header that can follow any hop-by-hop options header. The header is processed at the final destination and at each visited address specified by a routing header. Alternatively, the destination options header can follow any Encapsulating Security Payload (ESP) header. The destination options header is processed only at the final destination.
Routing header	43	Header that is used for source routing.
Fragment header	44	Header that is used when a source fragments a packet that is larger than the maximum transmission unit (MTU) for the path between itself and a destination. The Fragment header is used in each fragmented packet.
Upper-layer headers	6 (TCP) 17 (UDP)	Headers that are used inside a packet to transport the data. The two main transport protocols are TCP and UDP.

Note Some switch models support only a subset of IPv6 extension header types. The following list shows the extension header types that are supported by Cisco Nexus 3600 Platform Switches (N3K-C36180YC-R and N3K-C3636C-R) and by Cisco Nexus 9504 and 9508 modular chassis with these line cards: N9K-X9636C-R, N9K-X9636Q-R, N9K-X9636C-RX, and N9K-X96136YC-R.

Supported: Destination options (60), Routing (43), Fragment (44), Mobility (135), Host Identity Protocol (HIP) (139), Shim6 (140).

Not supported: Hop-by-hop options (0), Encapsulation Security Payload (50), Authentication Header (51), and experimental (253 and 254).

DNS for IPv6

IPv6 supports DNS record types that are supported in the DNS name-to-address and address-to-name lookup processes. The DNS record types support IPv6 addresses (see the following table).

Note IPv6 also supports the reverse mapping of IPv6 addresses to DNS names.

Table 7: IPv6 DNS Record Types

Record Type	Description	Format
AAAA	Maps a hostname to an IPv6 address. (Equivalent to an A record in IPv4.)	www.abc.test AAAA 3FFE:YYYY:C18:1::2
PTR	Maps an IPv6 address to a hostname. (Equivalent to a PTR record in IPv4.)	2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.0.0.0.8.1.c.0.y.y.y.y.e.f.f.3.ip6.int PTR www.abc.test

Path MTU Discovery for IPv6

As in IPv4, you can use path MTU discovery in IPv6 to allow a host to dynamically discover and adjust to differences in the MTU size of every link along a data path. In IPv6, however, fragmentation is handled by the source of a packet when the path MTU of one link along a given data path is not large enough to accommodate the size of the packets. Having IPv6 hosts handle packet fragmentation saves IPv6 router processing resources and helps IPv6 networks run more efficiently. Once the path MTU is reduced by the arrival of an ICMP Too Big message, Cisco NX-OS retains the lower value. The connection does not increase the segment size to gauge the throughput.

Note In IPv6, the minimum link MTU is 1280 octets. We recommend that you use an MTU value of 1500 octets for IPv6 links.

CDP IPv6 Address Support

You can use the Cisco Discovery Protocol (CDP) IPv6 address support for the neighbor information feature to transfer IPv6 addressing information between two Cisco devices. Cisco Discovery Protocol support for IPv6 addresses provides IPv6 information to network management products and troubleshooting tools.

ICMP for IPv6

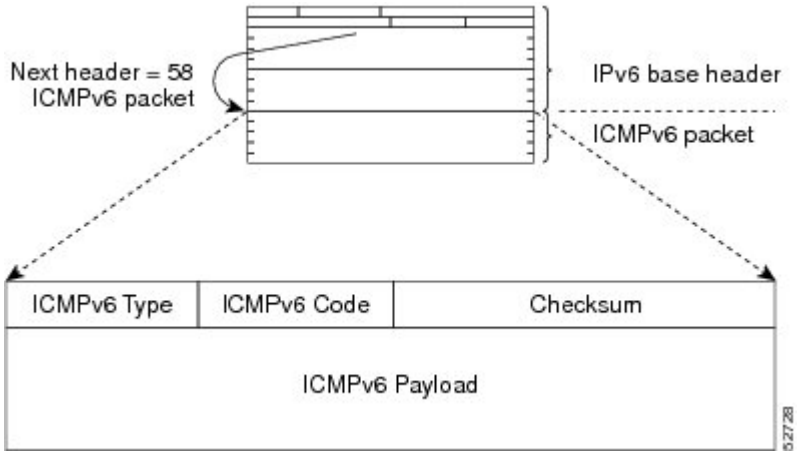
You can use ICMP in IPv6 to provide information about the health of the network. ICMPv6, the version that works with IPv6, reports errors if packets cannot be processed correctly and sends informational messages about the status of the network. For example, if a router cannot forward a packet because it is too large to be sent out on another network, the router sends out an ICMPv6 message to the originating host. Additionally, ICMP packets in IPv6 are used in IPv6 neighbor discovery and path MTU discovery. The path MTU discovery process ensures that a packet is sent using the largest possible size that is supported on a specific route.

A value of 58 in the Next Header field of the base IPv6 packet header identifies an IPv6 ICMP packet. The ICMP packet follows all the extension headers and is the last piece of information in the IPv6 packet.Within the IPv6 ICMP packets, the ICMPv6 Type and ICMPv6 Code fields identify IPv6 ICMP packet specifics, such as the ICMP message type. The value in the Checksum field is computed by the sender and checked by the receiver from the fields in the IPv6 ICMP packet and the IPv6 pseudo header.

Note The IPv6 header does not have a checksum. But a checksum on the transport layer can determine if packets have not been delivered correctly. All checksum calculations that include the IP address in the calculation must be modified for IPv6 to accommodate the new 128-bit address. A checksum is generated using a pseudo header.

The ICMPv6 Payload field contains error or diagnostic information that relates to IP packet processing. The following figure shows the IPv6 ICMP packet header format.

Figure 13: IPv6 ICMP Packet Header Format



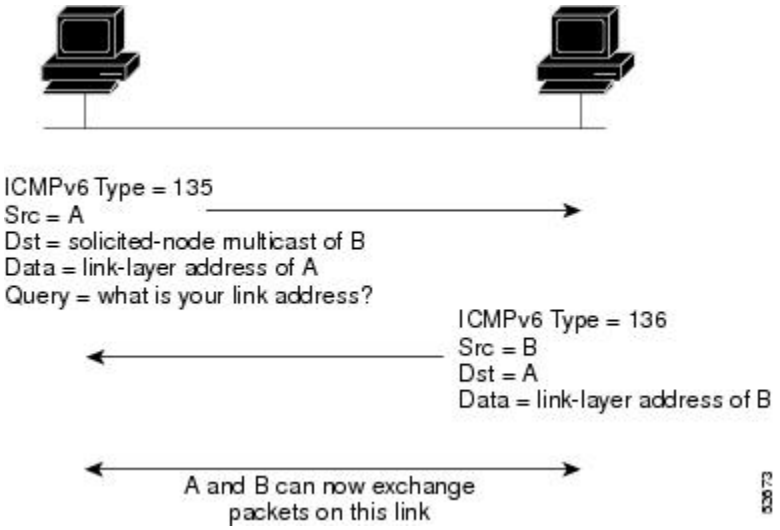
IPv6 Neighbor Discovery

You can use the IPv6 Neighbor Discovery Protocol (NDP) to determine whether a neighboring router is reachable. IPv6 nodes use neighbor discovery to determine the addresses of nodes on the same network (local link), to find neighboring routers that can forward their packets, to verify whether neighboring routers are reachable or not, and to detect changes to link-layer addresses. NDP uses ICMP messages to detect whether packets are sent to neighboring routers that are unreachable.

IPv6 Neighbor Solicitation Message

A node sends a neighbor solicitation message, which has a value of 135 in the Type field of the ICMP packet header, on the local link when it wants to determine the link-layer address of another node on the same local link (see the following figure). The source address is the IPv6 address of the node that sends the neighbor solicitation message. The destination address is the solicited-node multicast address that corresponds to the IPv6 address of the destination node. The neighbor solicitation message also includes the link-layer address of the source node.

Figure 14: IPv6 Neighbor Discovery—Neighbor Solicitation Message



After receiving the neighbor solicitation message, the destination node replies by sending a neighbor advertisement message, which has a value of 136 in the Type field of the ICMP packet header, on the local link. The source address is the IPv6 address of the node (the IPv6 address of the node interface that sends the neighbor advertisement message). The destination address is the IPv6 address of the node that sends the neighbor solicitation message. The data portion includes the link-layer address of the node that sends the neighbor advertisement message.

After the source node receives the neighbor advertisement, the source node and destination node can communicate.

Neighbor solicitation messages can verify the reachability of a neighbor after a node identifies the link-layer address of a neighbor. When a node wants to verify the reachability of a neighbor, it uses the destination address in a neighbor solicitation message as the unicast address of the neighbor.

Neighbor advertisement messages are also sent when there is a change in the link-layer address of a node on a local link. When there is a change, the destination address for the neighbor advertisement is the all-nodes multicast address.

Neighbor unreachability detection identifies the failure of a neighbor or the failure of the forward path to the neighbor and is used for all paths between hosts and neighboring nodes (hosts or routers). Neighbor unreachability detection is performed for neighbors to which only unicast packets are being sent and is not performed for neighbors to which multicast packets are being sent.

A neighbor is considered reachable when a positive acknowledgment is returned from the neighbor (indicating that packets previously sent to the neighbor have been received and processed). A positive acknowledgment—from an upper-layer protocol (such as TCP)—indicates that a connection is making forward progress (reaching its destination). If packets are reaching the peer, they are also reaching the next-hop neighbor of the source. Forward progress is also a confirmation that the next-hop neighbor is reachable.

For destinations that are not on the local link, forward progress implies that the first-hop router is reachable. When acknowledgments from an upper-layer protocol are not available, a node probes the neighbor using unicast neighbor solicitation messages to verify that the forward path is still working. The return of a solicited neighbor advertisement message from the neighbor is a positive acknowledgment that the forward path is still working (neighbor advertisement messages that have the solicited flag set to a value of 1 are sent only in response to a neighbor solicitation message). Unsolicited messages confirm only the one-way path from the source to the destination node; solicited neighbor advertisement messages indicate that a path is working in both directions.

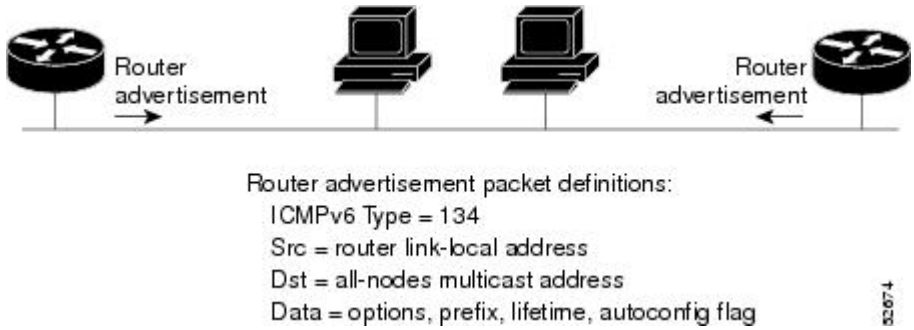
Note A neighbor advertisement message that has the solicited flag set to a value of 0 is not considered as a positive acknowledgment that the forward path is still working.

Router Advertisement Message

Router advertisement (RA) messages, which have a value of 134 in the Type field of the ICMP packet header, are periodically sent out to each configured interface of an IPv6 router.

The RA messages are sent to the all-nodes multicast address (see the following figure).

Figure 15: IPv6 Neighbor Discovery—RA Message



RA messages typically include the following information:

- One or more onlink IPv6 prefixes that nodes on the local link can use to automatically configure their IPv6 addresses
- Life-time information for each prefix included in the advertisement
- Default router information (whether the router sending the advertisement should be used as a default router and, if so, the amount of time in seconds that the router should be used as a default router)

- Additional information for hosts, such as the hop limit and MTU that a host should use in packets that it originates

RAs are also sent in response to router solicitation messages. Router solicitation messages, which have a value of 133 in the Type field of the ICMP packet header, are sent by hosts at system startup so that the host can immediately autoconfigure without needing to wait for the next scheduled RA message. The source address is usually the unspecified IPv6 address (0:0:0:0:0:0:0:0). If the host has a configured unicast address, the unicast address of the interface that sends the router solicitation message is used as the source address in the message. The destination address is the all-routers multicast address with a scope of the link. When an RA is sent in response to a router solicitation, the destination address in the RA message is the unicast address of the source of the router solicitation message.

You can configure the following RA message parameters:

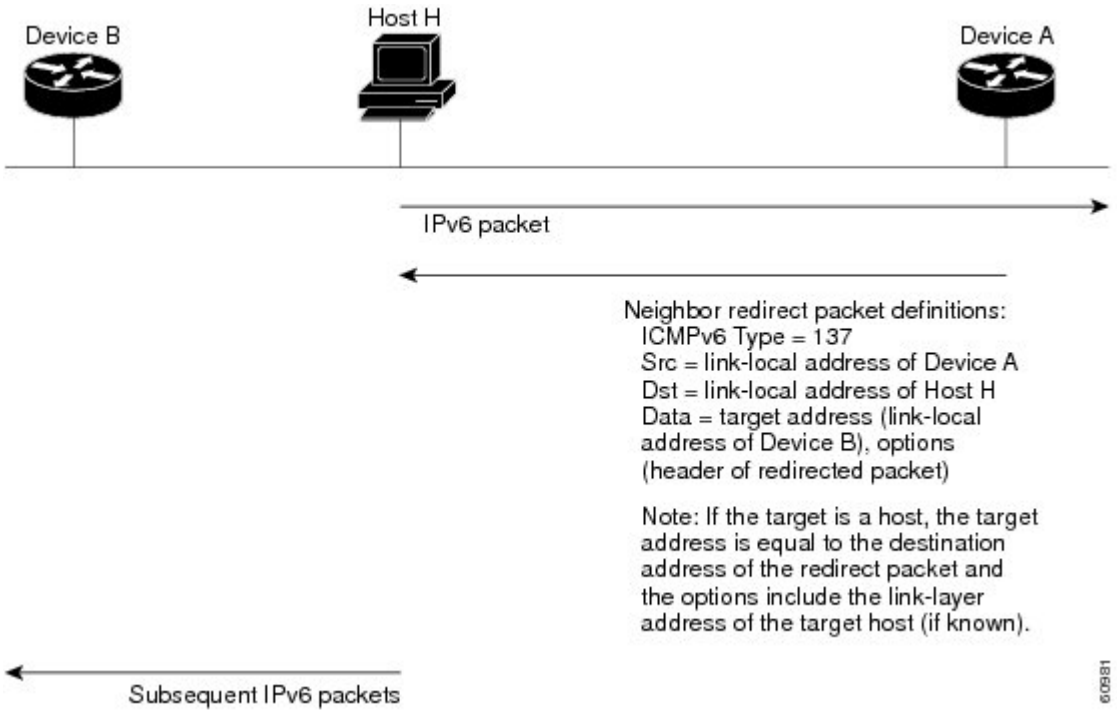
- The time interval between periodic RA messages
- The router life-time value, which indicates the usefulness of a router as the default router (for use by all nodes on a given link)
- The network prefixes in use on a given link
- The time interval between neighbor solicitation message retransmissions (on a given link)
- The amount of time that a node considers a neighbor reachable (for use by all nodes on a given link)

The configured parameters are specific to an interface. The sending of RA messages (with default values) is automatically enabled on Ethernet interfaces. For other interface types, you must enter the **no ipv6 nd suppress-ra** command to send RA messages. You can disable the RA message feature on individual interfaces by entering the **ipv6 nd suppress-ra** command.

IPv6 Neighbor Redirect Message

Routers send neighbor redirect messages to inform hosts of better first-hop nodes on the path to a destination (see the following figure). A value of 137 in the Type field of the ICMP packet header identifies an IPv6 neighbor redirect message.

Figure 16: IPv6 Neighbor Discovery—Neighbor Redirect Message



Note A router must be able to determine the link-local address for each of its neighboring routers in order to ensure that the target address (the final destination) in a redirect message identifies the neighbor router by its link-local address. For static routing, you should specify the address of the next-hop router using the link-local address of the router. For dynamic routing, you must configure all IPv6 routing protocols to exchange the link-local addresses of neighboring routers.

After forwarding a packet, a router sends a redirect message to the source of the packet under the following circumstances:

- The destination address of the packet is not a multicast address.
- The packet was not addressed to the router.
- The packet is about to be sent out the interface on which it was received.
- The router determines that a better first-hop node for the packet resides on the same link as the source of the packet.
- The source address of the packet is a global IPv6 address of a neighbor on the same link or a link-local address.

Virtualization Support

IPv6 supports virtual routing and forwarding (VRF) instances.

IPv6 Routes with ECMP

If all next-hops for a route are glean, drop, or punt, all next-hops are programmed as-is in the Multipath hardware table.

If some next-hops for a route are glean, drop, or punt, and the remaining next-hops are not, then only non glean, drop, or punt next-hops are programmed in the Multipath hardware table.

When a specific next-hop for ECMP route is resolved (ARP/IPV6 ND resolved), then the Multipath hardware table is updated accordingly.

Prerequisites for IPv6

IPv6 has the following prerequisites:

- You must be familiar with IPv6 basics such as IPv6 addressing, IPv6 header information, ICMPv6, and the IPv6 Neighbor Discovery (ND) Protocol.
- Ensure that you follow the memory/processing guidelines when you make a device a dual-stack device (IPv4/IPv6).

Guidelines and Limitations for IPv6

IPv6 has the following configuration guidelines and limitations:

- IPv6 packets are transparent to Layer 2 LAN switches because the switches do not examine Layer 3 packet information before forwarding IPv6 frames. IPv6 hosts can be directly attached to Layer 2 LAN switches.
- You can configure multiple IPv6 global addresses within the same prefix on an interface. However, multiple IPv6 link-local addresses on an interface are not supported.
- Because RFC 3879 deprecates the use of site-local addresses, you should configure private IPv6 addresses according to the recommendations of unique local addressing (ULA) in RFC 4193.

Default Settings

The following table lists the default settings for IPv6 parameters.

Table 8: Default IPv6 Parameters

Parameters	Default
ND reachable time	0 milliseconds
Neighbor solicitation retransmit interval	1000 milliseconds

Configuring IPv6

Configuring IPv6 Addressing

You must configure an IPv6 address on an interface so that the interface can forward IPv6 traffic. When you configure a global IPv6 address on an interface, it automatically configures a link-local address and activates IPv6 for that interface.

Procedure

	Command or Action	Purpose

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

Step 2	**interface ethernet number**
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3		

Option	Description
Command	Purpose
ipv6 address { *addr* [**eui64**] [**route-preference** *preference*] [**secondary**] **tag** *tag-id*]	Specifies an IPv6 address assigned to the interface and enables IPv6 processing on the interface. Entering the **ipv6 address** command configures global IPv6 addresses with an interface identifier (ID) in the low-order 64 bits of the IPv6 address. Only the 64-bit network prefix for the address needs to be specified; the last 64 bits are automatically computed from the interface ID.
ipv6 address ipv6-address use-link-local-only	Entering the **ipv6 address use-link-local-only** command configures a link-local address on the

	Command or Action		Purpose
	Option	**Description**	
		interface that is used instead of the link-local address that is automatically configured when IPv6 is enabled on the interface. This command enables IPv6 processing on an interface without configuring an IPv6 address.	

	Example:
	`switch(config-if)# ipv6 address 2001:0DB8::1/10`
	or
	`switch(config-if)# ipv6 address use-link-local-only`		

Step 4	(Optional) **show ipv6 interface** **Example:** `switch(config-if)# show ipv6 interface`		Displays interfaces configured for IPv6.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`		Saves this configuration change.

Example

This example shows how to configure an IPv6 address:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# ipv6 address ?
A:B::C:D/LEN IPv6 prefix format: xxxx:xxxx/ml, xxxx:xxxx::/ml,
xxxx::xx/128
use-link-local-only Enable IPv6 on interface using only a single link-local
address
switch(config-if)# ipv6 address 2001:db8::/64 eui64
```

This example shows how to display an IPv6 interface:

```
switch(config-if)# show ipv6 interface ethernet 3/1
Ethernet3/1, Interface status: protocol-down/link-down/admin-down, iod: 36
IPv6 address: 0dc3:0dc3:0000:0000:0218:baff:fed8:239d
IPv6 subnet: 0dc3:0dc3:0000:0000:0000:0000:0000:0000/64
IPv6 link-local address: fe80::0218:baff:fed8:239d (default)
IPv6 multicast routing: disabled
```

```
IPv6 multicast groups locally joined:
ff02::0001:ffd8:239d ff02::0002 ff02::0001 ff02::0001:ffd8:239d
IPv6 multicast (S,G) entries joined: none
IPv6 MTU: 1500 (using link MTU)
IPv6 RP inbound packet-filtering policy: none
IPv6 RP outbound packet-filtering policy: none
IPv6 inbound packet-filtering policy: none
IPv6 outbound packet-filtering policy: none
IPv6 interface statistics last reset: never
IPv6 interface RP-traffic statistics: (forwarded/originated/consumed)
Unicast packets: 0/0/0
Unicast bytes: 0/0/0
Multicast packets: 0/0/0
Multicast bytes: 0/0/0
```

Configuring IPv6 Neighbor Discovery

You can configure IPv6 neighbor discovery on the router. NDP enables IPv6 nodes and routers to determine the link-layer address of a neighbor on the same link, find neighboring routers, and keep track of neighbors.

Before you begin

You must first enable IPv6 on the interface.

Procedure

	Command or Action	Purpose

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

Step 2	**interface ethernet number**
	Example:
	`switch(config)# interface ethernet 2/31`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**ipv6 nd** [**hop-limit** *hop-limit* | **managed-config-flag** | **mtu** *mtu* | **ns-interval** *interval* | **other-config-flag** | **prefix** | **ra-interval** *interval* | **ra-lifetime** *lifetime* | **reachable-time** *time* | **redirects** | **retrans-timer** *time* | **suppress-ra**] **Example:** `switch(config-if)# ipv6 nd prefix`	Neighbor discovery is enabled automatically when you configure an IPv6 address. This command enables the following additional IPv6 neighbor discovery options on the interface: • **hop-limit** *hop-limit* —Advertises the hop limit in IPv6 neighbor discovery packets. The range is from 0 to 255. • **managed-config-flag** —Advertises in ICMPv6 router-advertisement messages to use stateful address auto configuration to obtain address information. • **mtu** *mtu* —Advertises the maximum transmission unit (MTU) in ICMPv6

	Command or Action	Purpose
		router-advertisement messages on this link. The range is from 1280 to 65535 bytes. • **ns-interval** *interval* —Configures the retransmission interval between IPv6 neighbor solicitation messages. The range is from 1000 to 3600000 milliseconds. • **other-config-flag** —Indicates in ICMPv6 router-advertisement messages that hosts use stateful auto configuration to obtain nonaddress related information. • **prefix** —Advertises the IPv6 prefix in the router-advertisement messages. • **ra-interval** *interval* —Configures the interval between sending ICMPv6 router-advertisement messages. The range is from 4 to 1800 seconds. • **ra-lifetime** *lifetime* —Advertises the lifetime of a default router in ICMPv6 router-advertisement messages. The range is from 0 to 9000 seconds. • **reachable-time** *time* —Advertises the time when a node considers a neighbor up after receiving a reachability confirmation in ICMPv6 router-advertisement messages. The range is from 0 to 9000 seconds. • **redirects** —Enables sending ICMPv6 redirect messages. • **retrans-timer** *time* —Advertises the time between neighbor-solicitation messages in ICMPv6 router-advertisement messages. The range is from 0 to 9000 seconds. • **suppress-ra** —Disables sending ICMPv6 router-advertisement messages.
Step 4	(Optional) **show ipv6 nd interface** **Example:** `switch(config-if)# show ipv6 nd interface`	Displays interfaces configured for IPv6 neighbor discovery.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure IPv6 neighbor discovery reachable time:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# ipv6 nd reachable-time 10
```

This example shows how to display an IPv6 neighbor discovery interface:

```
switch(config-if)# show ipv6 nd interface ethernet 3/1
ICMPv6 ND Interfaces for VRF "default"
Ethernet3/1, Interface status: protocol-down/link-down/admin-down
IPv6 address: 0dc3:0dc3:0000:0000:0218:baff:fed8:239d
ICMPv6 active timers:
Last Neighbor-Solicitation sent: never
Last Neighbor-Advertisement sent: never
Last Router-Advertisement sent:never
Next Router-Advertisement sent in: 0.000000
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: 10 ms
Send "Retrans Timer" field: 0 ms
Neighbor-Solicitation parameters:
NS retransmit interval: 1000 ms
ICMPv6 error message parameters:
Send redirects: false
Send unreachables: false
```

Optional IPv6 Neighbor Discovery

You can use the following optional IPv6 Neighbor Discovery commands:

Command	Purpose
ipv6 nd hop-limit	Configures the maximum number of hops used in router advertisements and all IPv6 packets that are originated by the router.
ipv6 nd managed-config-flag	Sets the managed address configuration flag in IPv6 router advertisements.
ipv6 nd mtu	Sets the maximum transmission unit (MTU) size of IPv6 packets sent on an interface.
ipv6 nd ns-interval	Configures the interval between IPv6 neighbor solicitation retransmissions on an interface.
ipv6 nd other-config-flag	Configures the other stateful configuration flag in IPv6 router advertisements.
ipv6 nd ra-interval	Configures the interval between IPv6 router advertisement (RA) transmissions on an interface.

Command	Purpose
ipv6 nd ra-lifetime	Configures the router lifetime value in IPv6 router advertisements on an interface.
ipv6 nd reachable-time	Configures the amount of time that a remote IPv6 node is considered reachable after some reachability confirmation event has occurred.
ipv6 nd redirects	Enables ICMPv6 redirect messages to be sent.
ipv6 nd retrans-timer	Configures the advertised time between neighbor solicitation messages in router advertisements.
ipv6 nd suppress-ra	Suppresses IPv6 router advertisement transmissions on a LAN interface.

Configuring IPv6 Packet Verification

Cisco NX-OS supports an Intrusion Detection System (IDS) that checks for IPv6 packet verification. You can enable or disable these IDS checks.

Note Cisco Nexus 3600 platform switches do not filter out packets with a source IP address of 0.0.0.0.

To enable IDS checks, use the following commands in global configuration mode:

Command	Purpose
hardware ip verify address { **destination zero** | **identical** | **reserved** | **source multicast** }	Performs the following IDS checks on the IPv6 address: • **destination zero** —Drops IPv6 packets if the destination IP address is ::. • **identical** —Drops IPv6 packets if the source IPv6 address is identical to the destination IPv6 address. • **reserved** —Drops IPv6 packets if the IPv6 address is ::1. • **source multicast** —Drops IPv6 packets if the IPv6 source address is in the FF00::/8 range (multicast).

Command	Purpose
hardware ipv6 verify length { **consistent** | **maximum** { max-frag | **max-tcp** | **udp** }}	Performs the following IDS checks on the IPv6 address: • **consistent** —Drops IPv6 packets where the Ethernet frame size is greater than or equal to the IPv6 packet length plus the Ethernet header. • **maximum max-frag** —Drops IPv6 packets if the formula (IPv6 Payload Length – IPv6 Extension Header Bytes) + (Fragment Offset * 8) is greater than 65536. • **maximum max-tcp** —Drops IPv6 packets if the TCP length is greater than the IP payload length. • **maximum udp** —Drops IPv6 packets if the IPv6 payload length is less than the UDP packet length.
hardware ipv6 verify tcp tiny-frag	Drops TCP packets if the IPv6 fragment offset is 1, or if the IPv6 fragment offset is 0 and the IP payload length is less than 16.
hardware ipv6 verify version	Drops IPv6 packets if the EtherType is not set to 6 (IPv6).

Use the **show hardware forwarding ip verify** command to display the IPv6 packet verification configuration.

Verifying the IPv6 Configuration

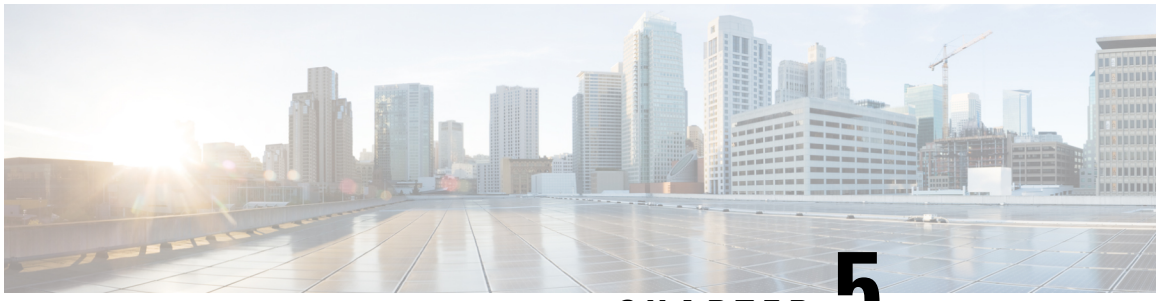
To display the IPv6 configuration, perform one of the following tasks:

Command	Purpose
show hardware forwarding ip verify	Displays the IPv4 and IPv6 packet verification configuration.
show ipv6 interface	Displays IPv6-related interface information.
show ipv6 adjacency	Displays the adjacency table.
show ipv6 icmp	Displays ICMPv6 information.
show ipv6 nd	Displays IPv6 neighbor discovery interface information.
show ipv6 neighbor	Displays IPv6 neighbor entry.

Configuration Examples for IPv6

This example shows how to configure IPv6:

```
configure terminal
interface ethernet 3/1
ipv6 address 2001:db8::/64 eui64
ipv6 nd reachable-time 10
```



CHAPTER 5

Configuring OSPFv2

This chapter describes how to configure Open Shortest Path First version 2 (OSPFv2) for IPv4 networks on Cisco NX-OS switches.

This chapter includes the following sections:

- Information About OSPFv2, on page 57
- Prerequisites for OSPFv2, on page 67
- Guidelines and Limitations, on page 67
- Default Settings, on page 67
- Configuring Basic OSPFv2, on page 68
- Configuring Advanced OSPFv2, on page 77
- Verifying the OSPFv2 Configuration, on page 94
- Displaying OSPFv2 Statistics, on page 95
- Configuration Examples for OSPFv2, on page 96
- Additional References, on page 96

Information About OSPFv2

OSPFv2 is an IETF link-state protocol (see the Link-State Protocols section) for IPv4 networks. An OSPFv2 router sends a special message, called a hello packet, out each OSPF-enabled interface to discover other OSPFv2 neighbor routers. Once a neighbor is discovered, the two routers compare information in the hello packet to determine if the routers have compatible configurations. The neighbor routers attempt to establish adjacency, which means that the routers synchronize their link-state databases to ensure that they have identical OSPFv2 routing information. Adjacent routers share link-state advertisements (LSAs) that include information about the operational state of each link, the cost of the link, and any other neighbor information. The routers then flood these received LSAs out every OSPF-enabled interface so that all OSPFv2 routers eventually have identical link-state databases. When all OSPFv2 routers have identical link-state databases, the network is converged (see the Convergence section). Each router then uses Dijkstra's Shortest Path First (SPF) algorithm to build its route table.

You can divide OSPFv2 networks into areas. Routers send most LSAs only within one area, which reduces the CPU and memory requirements for an OSPF-enabled router.

OSPFv2 supports IPv4.

Hello Packet

OSPFv2 routers periodically send hello packets on every OSPF-enabled interface. The hello interval determines how frequently the router sends these hello packets and is configured per interface. OSPFv2 uses hello packets for the following tasks:

- Neighbor discovery
- Keepalives
- Designated router election (see the Designated Routers section)

The hello packet contains information about the originating OSPFv2 interface and router, including the assigned OSPFv2 cost of the link, the hello interval, and optional capabilities of the originating router. An OSPFv2 interface that receives these hello packets determines if the settings are compatible with the receiving interface settings. Compatible interfaces are considered neighbors and are added to the neighbor table (see the Neighbors section).

Hello packets also include a list of router IDs for the routers that the originating interface has communicated with. If the receiving interface sees its own router ID in this list, then bidirectional communication has been established between the two interfaces.

OSPFv2 uses hello packets as a keepalive message to determine if a neighbor is still communicating. If a router does not receive a hello packet by the configured dead interval (usually a multiple of the hello interval), then the neighbor is removed from the local neighbor table.

Neighbors

An OSPFv2 interface must have a compatible configuration with a remote interface before the two can be considered neighbors. The two OSPFv2 interfaces must match the following criteria:

- Hello interval
- Dead interval
- Area ID (see the Areas section)
- Authentication
- Optional capabilities

If there is a match, the following information is entered into the neighbor table:

- Neighbor ID—The router ID of the neighbor.
- Priority—Priority of the neighbor. The priority is used for designated router election (see the Designated Routers section).
- State—Indication of whether the neighbor has just been heard from, is in the process of setting up bidirectional communications, is sharing the link-state information, or has achieved full adjacency.
- Dead time—Indication of the time since the last Hello packet was received from this neighbor.
- IP Address—The IP address of the neighbor.
- Designated Router—Indication of whether the neighbor has been declared as the designated router or as the backup designated router (see the Designated Routers section).

- Local interface—The local interface that received the hello packet for this neighbor.

Adjacency

Not all neighbors establish adjacency. Depending on the network type and designated router establishment, some neighbors become fully adjacent and share LSAs with all their neighbors, while other neighbors do not. For more information, see the Designated Routers section.

Adjacency is established using Database Description packets, Link State Request packets, and Link State Update packets in OSPF. The Database Description packet includes only the LSA headers from the link-state database of the neighbor (see theLink-State Database section). The local router compares these headers with its own link-state database and determines which LSAs are new or updated. The local router sends a Link State Request packet for each LSA that it needs new or updated information on. The neighbor responds with a Link State Update packet. This exchange continues until both routers have the same link-state information.

Designated Routers

Networks with multiple routers present a unique situation for OSPF. If every router floods the network with LSAs, the same link-state information will be sent from multiple sources. Depending on the type of network, OSPFv2 might use a single router, the designated router (DR), to control the LSA floods and represent the network to the rest of the OSPFv2 area (see the Areas section). If the DR fails, OSPFv2 selects a backup designated router (BDR). If the DR fails, OSPFv2 uses the BDR.

Network types are as follows:

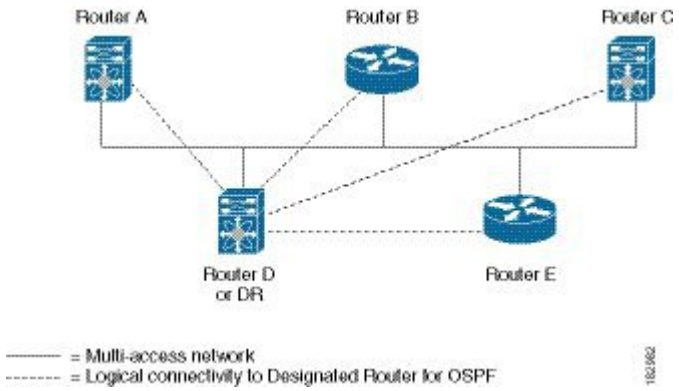
- Point-to-point—A network that exists only between two routers. All neighbors on a point-to-point network establish adjacency and there is no DR.
- Broadcast—A network with multiple routers that can communicate over a shared medium that allows broadcast traffic, such as Ethernet. OSPFv2 routers establish a DR and BDR that controls LSA flooding on the network. OSPFv2 uses the well-known IPv4 multicast addresses 224.0.0.5 and a MAC address of 0100.5300.0005 to communicate with neighbors.

The DR and BDR are selected based on the information in the Hello packet. When an interface sends a Hello packet, it sets the priority field and the DR and BDR field if it knows who the DR and BDR are. The routers follow an election procedure based on which routers declare themselves in the DR and BDR fields and the priority field in the Hello packet. As a final tie breaker, OSPFv2 chooses the highest router IDs as the DR and BDR.

All other routers establish adjacency with the DR and the BDR and use the IPv4 multicast address 224.0.0.6 to send LSA updates to the DR and BDR. Following figure shows this adjacency relationship between all routers and the DR.

DRs are based on a router interface. A router might be the DR for one network and not for another network on a different interface.

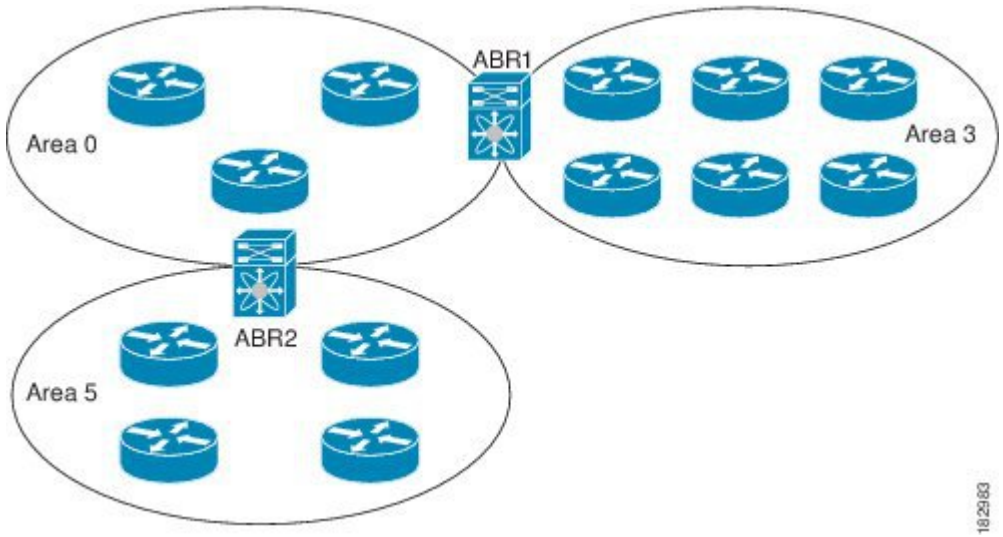
Figure 17: DR in Multi-Access Network



Areas

You can limit the CPU and memory requirements that OSPFv2 puts on the routers by dividing an OSPFv2 network into areas. An area is a logical division of routers and links within an OSPFv2 domain that creates separate subdomains. LSA flooding is contained within an area, and the link-state database is limited to links within the area. You can assign an area ID to the interfaces within the defined area. The Area ID is a 32-bit value that you can enter as a number or in dotted decimal notation, such as 10.2.3.1.Cisco NX-OS always displays the area in dotted decimal notation.If you define more than one area in an OSPFv2 network, you must also define the backbone area, which has the reserved area ID of 0. If you have more than one area, then one or more routers become area border routers (ABRs). An ABR connects to both the backbone area and at least one other defined area (see the following figure).

Figure 18: OSPFv2 Areas



The ABR has a separate link-state database for each area to which it connects. The ABR sends Network Summary (type 3) LSAs (see the Route Summarization section) from one connected area to the backbone area. The backbone area sends summarized information about one area to another area. In Figure **OSPFv2 Areas**, Area 0 sends summarized information about Area 5 to Area 3.

OSPFv2 defines one other router type: the autonomous system boundary router (ASBR). This router connects an OSPFv2 area to another autonomous system. An autonomous system is a network controlled by a single technical administration entity. OSPFv2 can redistribute its routing information into another autonomous system or receive redistributed routes from another autonomous system. For more information, see Advanced Features section).

Link-State Advertisements

OSPFv2 uses link-state advertisements (LSAs) to build its routing table.

LSA Types

The following table shows the LSA types supported by Cisco NX-OS.

Table 9: LSA types

Type	Name	Description
1	Router LSA	LSA sent by every router. This LSA includes the state and the cost of all links and a list of all OSPFv2 neighbors on the link. Router LSAs trigger an SPF recalculation. Router LSAs are flooded to local OSPFv2 area. See the Areas section.
2	Network LSA	LSA sent by the DR. This LSA lists all routers in the multi-access network. Network LSAs trigger an SPF recalculation. See the Designated Routers section.
3	Network Summary LSA	LSA sent by the area border router to an external area for each destination in the local area. This LSA includes the link cost from the area border router to the local destination. See the Areas section.
4	ASBR Summary LSA	LSA sent by the area border router to an external area. This LSA advertises the link cost to the ASBR only. See the Areas section.
5	AS External LSA	LSA generated by the ASBR. This LSA includes the link cost to an external autonomous system destination. AS External LSAs are flooded throughout the autonomous system. See the Areas section.

Type	Name	Description
7	NSSA External LSA	LSA generated by the ASBR within a not-so-stubby area (NSSA). This LSA includes the link cost to an external autonomous system destination. NSSA External LSAs are flooded only within the local NSSA. See the Areas section.
9-11	Opaque LSAs	LSA used to extend OSPF. See the Opaque LSAs section.

Link Cost

Each OSPFv2 interface is assigned a link cost. The cost is an arbitrary number. By default, Cisco NX-OS assigns a cost that is the configured reference bandwidth divided by the interface bandwidth. By default, the reference bandwidth is 40 Gb/s. The link cost is carried in the LSA updates for each link.

Flooding and LSA Group Pacing

When an OSPFv2 router receives an LSA, it forwards that LSA out every OSPF-enabled interface, flooding the OSPFv2 area with this information. This LSA flooding guarantees that all routers in the network have identical routing information. LSA flooding depends on the OSPFv2 area configuration (see the Areas section). The LSAs are flooded based on the link-state refresh time (every 30 minutes by default). Each LSA has its own link-state refresh time.

You can control the flooding rate of LSA updates in your network by using the LSA group pacing feature. LSA group pacing can reduce high CPU or buffer utilization. This feature groups LSAs with similar link-state refresh times to allow OSPFv2 to pack multiple LSAs into an OSPFv2 Update message.

By default, LSAs with link-state refresh times within four minutes of each other are grouped together. You should lower this value for large link-state databases or raise it for smaller databases to optimize the OSPFv2 load on your network.

Link-State Database

Each router maintains a link-state database for the OSPFv2 network. This database contains all the collected LSAs, and includes information on all the routes through the network. OSPFv2 uses this information to calculate the bast path to each destination and populates the routing table with these best paths.

LSAs are removed from the link-state database if no LSA update has been received within a set interval, called the MaxAge. Routers flood a repeat of the LSA every 30 minutes to prevent accurate link-state information from being aged out. Cisco NX-OS supports the LSA grouping feature to prevent all LSAs from refreshing at the same time. For more information, see the Flooding and LSA Group Pacing section.

Opaque LSAs

Opaque LSAs allow you to extend OSPF functionality. Opaque LSAs consist of a standard LSA header followed by application-specific information. This information might be used by OSPFv2 or by other applications. Three Opaque LSA types are defined as follows:

- LSA type 9—Flooded to the local network.

- LSA type 10—Flooded to the local area.
- LSA type 11—Flooded to the local autonomous system.

OSPFv2 and the Unicast RIB

OSPFv2 runs the Dijkstra shortest path first algorithm on the link-state database. This algorithm selects the best path to each destination based on the sum of all the link costs for each link in the path. The resultant shortest path for each destination is then put in the OSPFv2 route table. When the OSPFv2 network is converged, this route table feeds into the unicast RIB. OSPFv2 communicates with the unicast RIB to do the following:

- Add or remove routes
- Handle route redistribution from other protocols
- Provide convergence updates to remove stale OSPFv2 routes and for stub router advertisements (see the OSPFv2 Stub Router Advertisements section.)

OSPFv2 also runs a modified Dijkstra algorithm for fast recalculation for summary and external (type 3, 4, 5, and 7) LSA changes.

Authentication

You can configure authentication on OSPFv2 messages to prevent unauthorized or invalid routing updates in your network. Cisco NX-OS supports two authentication methods:

- Simple password authentication
- MD5 authentication digest

You can configure the OSPFv2 authentication for an OSPFv2 area or per interface.

Simple Password Authentication

Simple password authentication uses a simple cleartext password that is sent as part of the OSPFv2 message. The receiving OSPFv2 router must be configured with the same cleartext password to accept the OSPFv2 message as a valid route update. Because the password is in cleartext, anyone who can watch traffic on the network can learn the password.

Cryptographic Authentication

Cryptographic authentication uses an encrypted password for OSPFv2 authentication. The transmitter computes a code using the packet to be transmitted and the key string, inserts the code and the key ID in the packet, and transmits the packet. The receiver validates the code in the packet by computing the code locally using the received packet and the key string (corresponding to the key ID in the packet) configured locally.

Both message digest 5 (MD5) and hash-based message authentication code secure hash algorithm (HMAC-SHA) cryptographic authentication are supported.

MD5 Authentication

You should use MD5 authentication to authenticate OSPFv2 messages. You configure a password that is shared at the local router and all remote OSPFv2 neighbors. For each OSPFv2 message, Cisco NX-OS creates

an MD5 one-way message digest based on the message itself and the encrypted password. The interface sends this digest with the OSPFv2 message. The receiving OSPFv2 neighbor validates the digest using the same encrypted password. If the message has not changed, the digest calculation is identical and the OSPFv2 message is considered valid.

MD5 authentication includes a sequence number with each OSPFv2 message to ensure that no message is replayed in the network.

Advanced Features

Cisco NX-OS supports a number of advanced OSPFv2 features that enhance the usability and scalability of OSPFv2 in the network.

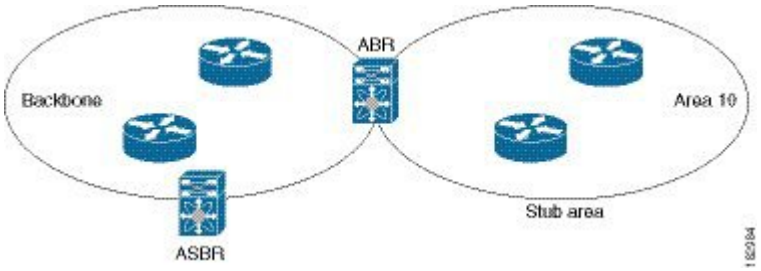
Stub Area

You can limit the amount of external routing information that floods an area by making it a stub area. A stub area is an area that does not allow AS External (type 5) LSAs (see the Link-State Advertisements section). These LSAs are usually flooded throughout the local autonomous system to propagate external route information. Stub areas have the following requirements:

- All routers in the stub area are stub routers. See the Stub Area section.
- No ASBR routers exist in the stub area.
- You cannot configure virtual links in the stub area.

The following figure shows an example of an OSPFv2 autonomous system where all routers in area 0.0.0.10 have to go through the ABR to reach external autonomous systems. area 0.0.0.10 can be configured as a stub area.

Figure 19: Stub Area



Stub areas use a default route for all traffic that needs to go through the backbone area to the external autonomous system. The default route is 0.0.0.0 for IPv4.

Not-So-Stubby Area

A Not-so-Stubby Area (NSSA) is similar to a stub area, except that an NSSA allows you to import autonomous system external routes within an NSSA using redistribution. The NSSA ASBR redistributes these routes and generates NSSA External (type 7) LSAs that it floods throughout the NSSA. You can optionally configure the ABR that connects the NSSA to other areas to translate this NSSA External LSA to AS External (type 5) LSAs. The area border router (ABR) then floods these AS External LSAs throughout the OSPFv2 autonomous system. Summarization and filtering are supported during the translation. See the Link-State Advertisements section for details on NSSA External LSAs.

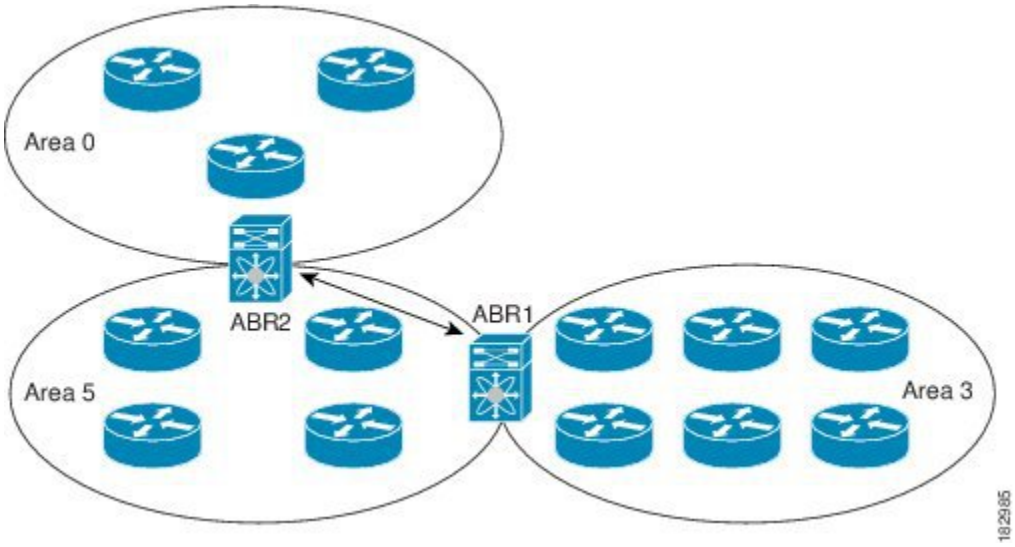
You can, for example, use NSSA to simplify administration if you are connecting a central site using OSPFv2 to a remote site that is using a different routing protocol. Before NSSA, the connection between the corporate site border router and a remote router could not be run as an OSPFv2 stub area because routes for the remote site could not be redistributed into a stub area. With NSSA, you can extend OSPFv2 to cover the remote connection by defining the area between the corporate router and remote router as an NSSA (see the Configuring NSSA section).

The backbone Area 0 cannot be an NSSA.

Virtual Links

Virtual links allow you to connect an OSPFv2 area ABR to a backbone area ABR when a direct physical connection is not available. The following figure shows a virtual link that connects Area 3 to the backbone area through Area 5.

Figure 20: Virtual Links



You can also use virtual links to temporarily recover from a partitioned area, which occurs when a link within the area fails, isolating part of the area from reaching the designated ABR to the backbone area.

Route Redistribution

OSPFv2 can learn routes from other routing protocols by using route redistribution. See the Route Redistribution section. You configure OSPFv2 to assign a link cost for these redistributed routes or a default link cost for all redistributed routes.

Route redistribution uses route maps to control which external routes are redistributed. See Configuring Route Policy Manager, for details on configuring route maps. You can use route maps to modify parameters in the AS External (type 5) and NSSA External (type 7) LSAs before these external routes are advertised in the local OSPFv2 autonomous system.

Route Summarization

Because OSPFv2 shares all learned routes with every OSPF-enabled router, you might want to use route summarization to reduce the number of unique routes that are flooded to every OSPF-enabled router. Route summarization simplifies route tables by replacing more-specific addresses with an address that represents

all the specific addresses. For example, you can replace 10.1.1.0/24, 10.1.2.0/24, and 10.1.3.0/24 with one summary address, 10.1.0.0/16.

Typically, you would summarize at the boundaries of area border routers (ABRs). Although you could configure summarization between any two areas, it is better to summarize in the direction of the backbone so that the backbone receives all the aggregate addresses and injects them, already summarized, into other areas. The two types of summarization are as follows:

- Inter-area route summarization
- External route summarization

You configure inter-area route summarization on ABRs, summarizing routes between areas in the autonomous system. To take advantage of summarization, you should assign network numbers in areas in a contiguous way to be able to lump these addresses into one range.

External route summarization is specific to external routes that are injected into OSPFv2 using route redistribution. You should make sure that external ranges that are being summarized are contiguous. Summarizing overlapping ranges from two different routers could cause packets to be sent to the wrong destination. Configure external route summarization on ASBRs that are redistributing routes into OSPF.

When you configure a summary address, Cisco NX-OS automatically configures a discard route for the summary address to prevent routing black holes and route loops.

OSPFv2 Stub Router Advertisements

You can configure an OSPFv2 interface to act as a stub router using the OSPFv2 stub router advertisements feature. Use this feature when you want to limit the OSPFv2 traffic through this router, such as when you want to introduce a new router to the network in a controlled manner or limit the load on a router that is already overloaded. You might also want to use this feature for various administrative or traffic engineering reasons.

OSPFv2 stub router advertisements do not remove the OSPFv2 router from the network topology, but they do prevent other OSPFv2 routers from using this router to route traffic to other parts of the network. Only the traffic that is destined for this router or directly connected to this router is sent.

OSPFv2 stub router advertisements mark all stub links (directly connected to the local router) to the cost of the local OSPFv2 interface. All remote links are marked with the maximum cost (0xFFFF).

Multiple OSPFv2 Instances

Cisco NX-OS supports multiple instances of the OSPFv2 protocol that run on the same node. You cannot configure multiple instances over the same interface. By default, every instance uses the same system router ID. You must manually configure the router ID for each instance if the instances are in the same OSPFv2 autonomous system.

SPF Optimization

Cisco NX-OS optimizes the SPF algorithm in the following ways:

- Partial SPF for Network (type 2) LSAs, Network Summary (type 3) LSAs, and AS External (type 5) LSAs—When there is a change on any of these LSAs, Cisco NX-OS performs a faster partial calculation rather than running the whole SPF calculation.
- SPF timers—You can configure different timers for controlling SPF calculations. These timers include exponential backoff for subsequent SPF calculations. The exponential backoff limits the CPU load of multiple SPF calculations.

BFD

This feature supports bidirectional forwarding detection (BFD). BFD is a detection protocol that provides fast forwarding-path failure detection times. BFD provides subsecond failure detection between two adjacent devices and can be less CPU-intensive than protocol hello messages because some of the BFD load can be distributed onto the data plane on supported modules.

Virtualization Support

OSPFv2 supports Virtual Routing and Forwarding (VRF) instances. By default, Cisco NX-OS places you in the default VRF unless you specifically configure another VRF. Each OSPFv2 instance can support multiple VRFs, up to the system limit. For more information, see Configuring Layer 3 Virtualization.

Prerequisites for OSPFv2

OSPFv2 has the following prerequisites:

- You must be familiar with routing fundamentals to configure OSPF.
- You are logged on to the switch.
- You have configured at least one interface for IPv4 that is capable of communicating with a remote OSPFv2 neighbor.
- You have installed the LAN Base Services license.
- You have completed the OSPFv2 network strategy and planning for your network. For example, you must decide whether multiple areas are required.
- You have enabled the OSPF feature (see the Enabling OSPFv2 section).

Guidelines and Limitations

OSPFv2 has the following configuration guidelines and limitations:

- Cisco NX-OS displays areas in dotted decimal notation regardless of whether you enter the area in decimal or dotted decimal notation.

Default Settings

The following table lists the default settings for OSPFv2 parameters.

Table 10: Default OSPFv2 Parameters

Parameters	Default
Hello interval	10 seconds
Dead interval	40 seconds

Parameters	Default
OSPFv2 feature	Disabled
Stub router advertisement announce time	600 seconds
Reference bandwidth for link cost calculation	40 Gb/s
LSA minimal arrival time	1000 milliseconds
LSA group pacing	240 seconds
SPF calculation initial delay time	0 milliseconds
SPF calculation hold time	5000 milliseconds
SPF calculation initial delay time	0 milliseconds

Configuring Basic OSPFv2

Configure OSPFv2 after you have designed your OSPFv2 network.

Enabling OSPFv2

You must enable the OSPFv2 feature before you can configure OSPFv2.

Procedure

	Command or Action	Purpose

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

Step 2	**feature ospf** **Example:** `switch(config)# feature ospf` **Example:**	Enables the OSPFv2 feature.
Step 3	(Optional) **show feature** **Example:** `switch(config)# show feature`	Displays enabled and disabled features.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

Use the **no feature ospf** command to disable the OSPFv2 feature and remove all associated configurations.

Command	Purpose
no feature ospf Example: `switch(config)# no feature ospf`	Disables the OSPFv2 feature and removes all associated configurations.

Creating an OSPFv2 Instance

The first step in configuring OSPFv2 is to create an OSPFv2 instance. You assign a unique instance tag for this OSPFv2 instance. The instance tag can be any string.

For more information about OSPFv2 instance parameters, see the Configuring Advanced OSPFv2 section.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Use the **show ip ospf** *instance-tag* command to verify that the instance tag is not in use.

OSPFv2 must be able to obtain a router identifier (for example, a configured loopback address) or you must configure the router ID option.

Procedure

	Command or Action	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	(Optional) **router-id** *ip-address* **Example:** `switch(config-router)# router-id 192.0.2.1`	Configures the OSPFv2 router ID. This IP address identifies this OSPFv2 instance and must exist on a configured interface in the system.
Step 4	(Optional) **show ip ospf** *instance-tag* **Example:** `switch(config-router)# show ip ospf 201`	Displays OSPF information.

	Command or Action	Purpose
Step 5	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

Use the **no router ospf** command to remove the OSPFv2 instance and all associated configurations.

Command	Purpose
no router ospf *instance-tag* Example: `switch(config)# no router ospf 201`	Deletes the OSPF instance and the associated configurations.

Note This command does not remove OSPF configuration in interface mode. You must manually remove any OSPFv2 commands configured in interface mode.

Configuring Optional Parameters on an OSPFv2 Instance

You can configure optional parameters for OSPF.

For more information about OSPFv2 instance parameters, see the Configuring Advanced OSPFv2 section.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

OSPFv2 must be able to obtain a router identifier (for example, a configured loopback address) or you must configure the router ID option.

Procedure

	Command or Action	Purpose
Step 1	**distance** *number* **Example:** `switch(config-router)# distance 25`	Configures the administrative distance for this OSPFv2 instance. The range is from 1 to 255. The default is 110.
Step 2	**log-adjacency-changes** [**detail**] **Example:** `switch(config-router)# log-adjacency-changes`	Generates a system message whenever a neighbor changes state.

	Command or Action	Purpose
Step 3	**maximum-paths** *path-number* **Example:** `switch(config-router)# maximum-paths 4`	Configures the maximum number of equal OSPFv2 paths to a destination in the route table. This command is used for load balancing. The range is from 1 to 16. The default is 8.

Example

This example shows how to create an OSPFv2 instance:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# copy running-config startup-config
```

Configuring Networks in OSPFv2

You can configure a network to OSPFv2 by associating it through the interface that the router uses to connect to that network (see the Neighbors section). You can add all networks to the default backbone area (Area 0), or you can create new areas using any decimal number or an IP address.

Note All areas must connect to the backbone area either directly or through a virtual link.

Note OSPF is not enabled on an interface until you configure a valid IP address for that interface.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Procedure

	Command or Action	Purpose

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

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

	Command or Action	Purpose
Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Enters global configuration mode.
Step 4	**ip address** *ip-prefix/length* **Example:** `switch(config-if)# ip address 192.0.2.1/16`	Assigns an IP address and subnet mask to this interface.
Step 5	**ip router ospf** *instance-tag* **area** *area-id* [**secondaries none**] **Example:** `switch(config-if)# ip router ospf 201 area 0.0.0.15`	Adds the interface to the OSPFv2 instance and area.
Step 6	(Optional) **show ip ospf** *instance-tag* **interface** *interface-type slot/port* **Example:** `switch(config-if)# show ip ospf 201 interface ethernet 1/2`	Displays OSPF information.
Step 7	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

You can configure the following optional parameters for OSPFv2 in interface configuration mode:

Command	Purpose
ip ospf cost *number* Example: `switch(config-if)# ip ospf cost 25`	Configures the OSPFv2 cost metric for this interface. The default is to calculate cost metric, based on reference bandwidth and interface bandwidth. The range is from 1 to 65535.
ip ospf dead-interval *seconds* Example: `switch(config-if)# ip ospf dead-interval 50`	Configures the OSPFv2 dead interval, in seconds. The range is from 1 to 65535. The default is four times the hello interval, in seconds.
ip ospf hello-interval *seconds* Example: `switch(config-if)# ip ospf hello-interval 25`	Configures the OSPFv2 hello interval, in seconds. The range is from 1 to 65535. The default is 10 seconds.

Command	Purpose
ip ospf mtu-ignore Example: `switch(config-if)# ip ospf mtu-ignore`	Configures OSPFv2 to ignore any IP MTU mismatch with a neighbor. The default is to not establish adjacency if the neighbor MTU does not match the local interface MTU.
ip ospf passive-interface Example: `switch(config-if)# ip ospf passive-interface`	Suppresses routing updates on the interface.
ip ospf priority *number* Example: `switch(config-if)# ip ospf priority 25`	Configures the OSPFv2 priority, used to determine the DR for an area. The range is from 0 to 255. The default is 1. See the Designated Routers section.
ip ospf shutdown Example: `switch(config-if)# ip ospf shutdown`	Shuts down the OSPFv2 instance on this interface.

This example shows how to add a network area 0.0.0.10 in OSPFv2 instance 201:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# ip address 192.0.2.1/16
switch(config-if)# ip router ospf 201 area 0.0.0.10
switch(config-if)# copy running-config startup-config
```

Use the **show ip ospf interface** command to verify the interface configuration. Use the **show ip ospf neighbor** command to see the neighbors for this interface.

Configuring Authentication for an Area

You can configure authentication for all networks in an area or for individual interfaces in the area. Interface authentication configuration overrides area authentication.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Ensure that all neighbors on an interface share the same authentication configuration, including the shared authentication key.

Create the keychain for this authentication configuration.

Procedure

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

	Command or Action	Purpose
	`switch# configure terminal` `switch(config)#`	

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**area** *area-id* **authentication** [**message-digest**] **Example:** `switch(config-router)# area 0.0.0.10 authentication`	Configures the authentication mode for an area.

Step 4	**interface** *interface-type slot/port*
	Example:
	`switch(config-router)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 5	**no switchport** **Example:** `switch(config-if)# no switchport`	Enters global configuration mode.
Step 6	(Optional) **ip ospf authentication-key** [**0** | **3**] *key* **Example:** `switch(config-if)# ip ospf authentication-key 0 mypass`	(Optional) Configures simple password authentication for this interface. Use this command if the authentication is not set to keychain or message-digest. 0 configures the password in cleartext. 3 configures the password as 3DES encrypted.
Step 7	(Optional) **ip ospf message-digest-key** *key-id* **md5** [**0** | **3**] *key* **Example:** `switch(config-if)# ip ospf message-digest-key 21 md5 0 mypass`	Configures message digest authentication for this interface. Use this command if the authentication is set to message-digest. The key-id range is from 1 to 255. The MD5 option 0 configures the password in cleartext and 3 configures the pass key as 3DES encrypted.
Step 8	(Optional) **show ip ospf** *instance-tag* **interface** *interface-type slot/port* **Example:** `switch(config-if)# show ip ospf 201 interface ethernet 1/2`	
Step 9	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Configuring Authentication for an Interface

You can configure authentication for individual interfaces in the area. Interface authentication configuration overrides area authentication.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Ensure that all neighbors on an interface share the same authentication configuration, including the shared authentication key.

Create the keychain for this authentication configuration.

To configure OSPFv2 HMAC-SHA authentication, you must specify the HMAC-SHA algorithm to be used for the key. OSPFv2 will use the MD5 cryptographic algorithm if cryptographic authentication using keychain is configured without selecting a cryptographic-algorithm.

Procedure

	Command or Action	Purpose

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

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

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

Step 4	**ip ospf authentication** [**message-digest**]
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enables interface authentication mode for OSPFv2 for either cleartext or message-digest type. Overrides area-based authentication for this interface. All neighbors must share this authentication type.

Step 5	**ip ospf authentication keychain** *key-name* **Example:** `switch(config-if)# ip ospf authentication keychain Test1`	Configures interface authentication to use keychains for OSPFv2.
Step 6	**ip ospf authentication-key** [**0** | **3** | **7**] *key* **Example:** `switch(config-if)# ip ospf authentication-key 0 mypass`	Configures simple password authentication for this interface. Use this command if the authentication is not set to keychain or message-digest. The options are as follows:

	Command or Action	Purpose
		• 0—Configures the password in cleartext. • 3—Configures the pass key as 3DES encrypted. • 7—Configures the key as Cisco type 7 encrypted.
Step 7	**ip ospf message-digest-key** *key-id* **md5** [**0** | **3** | **7**] *key* **Example:** `switch(config-if)# ip ospf message-digest-key 21 md5 0 mypass`	Configures message digest authentication for this interface. Use this command if the authentication is set to message-digest.The key-id range is from 1 to 255. The MD5 options are as follows: • 0—Configures the password in cleartext. • 3—Configures the pass key as 3DES encrypted. • 7—Configures the key as Cisco type 7 encrypted.
Step 8	**show ip ospf** *instance-tag* **interface** *interface-type slot/port* **Example:** `switch(config-if)# show ip ospf 201 interface ethernet 1/2`	Displays OSPF information.
Step 9	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to set an interface for simple, unencrypted passwords and set the password for Ethernet interface 1/2:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# exit
switch(config)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# ip router ospf 201 area 0.0.0.10
switch(config-if)# ip ospf authentication
switch(config-if)# ip ospf authentication-key 0 mypass
switch(config-if)# copy running-config startup-config
```

This example shows how to configure OSPFv2 HMAC-SHA-1 and MD5 cryptographic authentication:

```
switch# configure terminal
switch(config)# key chain chain1
switch(config-keychain)# key 1
```

```
switch(config-keychain-key)# key-string 7 070724404206
switch(config-keychain-key)# accept-lifetime 01:01:01 Jan 01 2015 infinite
switch(config-keychain-key)# send-lifetime 01:01:01 Jan 01 2015 infinite
switch(config-keychain-key)# cryptographic-algorithm HMAC-SHA-1 switch(config-keychain-key)#
 exit
switch(config-keychain)# key 2
switch(config-keychain-key)# key-string 7 070e234f1f5b4a
switch(config-keychain-key)# accept-lifetime 10:51:01 Jul 24 2015 infinite
switch(config-keychain-key)# send-lifetime 10:51:01 Jul 24 2015 infinite
switch(config-keychain-key)# cryptographic-algorithm MD5
switch(config-keychain-key)# exit
switch(config-keychain)# exit

switch(config)# interface ethernet 1/1
switch(config-if)# ip router ospf 1 area 0.0.0.0
switch(config-if)# ip ospf authentication message-digest
switch(config-if)# ip ospf authentication key-chain chain1

switch(config-if)# show key chain chain1
Key-Chain chain1
Key 1 -- text 7 "070724404206"
cryptographic-algorithm HMAC-SHA-1
accept lifetime UTC (01:01:01 Jan 01 2015)-(always valid) [active]
send lifetime UTC (01:01:01 Jan 01 2015)-(always valid) [active]
Key 2 -- text 7 "070e234f1f5b4a"
cryptographic-algorithm MD5
accept lifetime UTC (10:51:00 Jul 24 2015)-(always valid) [active]
send lifetime UTC (10:51:00 Jul 24 2015)-(always valid) [active]

switch(config-if)# show ip ospf interface ethernet 1/1
Ethernet1/1 is up, line protocol is up
IP address 11.11.11.1/24
Process ID 1 VRF default, area 0.0.0.3
Enabled by interface configuration
State BDR, Network type BROADCAST, cost 40
Index 6, Transmit delay 1 sec, Router Priority 1
Designated Router ID: 33.33.33.33, address: 11.11.11.3
Backup Designated Router ID: 1.1.1.1, address: 11.11.11.1
2 Neighbors, flooding to 2, adjacent with 2
Timer intervals: Hello 10, Dead 40, Wait 40, Retransmit 5
Hello timer due in 00:00:08
Message-digest authentication, using keychain key1 (ready)
Sending SA: Key id 2, Algorithm MD5
Number of opaque link LSAs: 0, checksum sum 0
```

Configuring Advanced OSPFv2

Configure OSPFv2 after you have designed your OSPFv2 network.

Configuring Filter Lists for Border Routers

You can separate your OSPFv2 domain into a series of areas that contain related networks. All areas must connect to the backbone area through an area border router (ABR). OSPFv2 domains can connect to external domains through an autonomous system border router (ASBR). See the Areas section.

ABRs have the following optional configuration parameters:

- Area range—Configures route summarization between areas.

- Filter list—Filters the Network Summary (type 3) LSAs on an ABR that are allowed in from an external area.

ASBRs also support filter lists.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Create the route map that the filter list uses to filter IP prefixes in incoming or outgoing Network Summary (type 3) LSAs. See Configuring Route Policy Manager.

Procedure

	Command or Action	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**area** *area-id* **filter-list route-map** *map-name* {**in** | **out**} **Example:** `switch(config-router)# area 0.0.0.10 filter-list route-map FilterLSAs in`	Filters incoming or outgoing Network Summary (type 3) LSAs on an ABR.

Step 4	(Optional) **show ip ospf policy statistics area** *id* **filter-list** {**in** | **out**}
	Example:
	`switch(config-router)# show ip ospf policy`
	`statistics area 0.0.0.10 filter-list in`
		Displays OSPF policy information.

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

Example

This example shows how to configure a filter list in area 0.0.0.10:

```
switch# configure terminal
switch(config)# router ospf 201
```

```
switch(config-router)# area 0.0.0.10 filter-list route-map FilterLSAs in
switch(config-router)# copy running-config startup-config
```

Configuring Stub Areas

You can configure a stub area for part of an OSPFv2 domain where external traffic is not necessary. Stub areas block AS External (type 5) LSAs and limit unnecessary routing to and from selected networks. See the Stub Area section. You can optionally block all summary routes from going into the stub area.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Ensure that there are no virtual links or ASBRs in the proposed stub area.

Procedure

	Command or Action	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**area** *area-id* **stub** **Example:** `switch(config-router)# area 0.0.0.10 stub`	Creates this area as a stub area.
Step 4	(Optional) **area** *area-id* **default-cost** *cost* **Example:** `switch(config-router)# area 0.0.0.10 default-cost 25`	Sets the cost metric for the default summary route sent into this stub area. The range is from 0 to 16777215. The default is 1.
Step 5	(Optional) **show ip ospf** *instance-tag* **Example:** `switch(config-router)# show ip ospf 201`	Displays OSPF information.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Example

This example shows how to create a stub area:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# area 0.0.0.10 stub
switch(config-router)# copy running-config startup-config
```

Configuring a Totally Stubby Area

You can create a totally stubby area and prevent all summary route updates from going into the stub area.

To create a totally stubby area, use the following command in router configuration mode:

Command	Purpose
area *area-id* stub no-summary Example: `switch(config-router)# area 20 stub no-summary`	Creates this area as a totally stubby area.

Configuring NSSA

You can configure an NSSA for part of an OSPFv2 domain where limited external traffic is required. See the Not-So-Stubby Area section. You can optionally translate this external traffic to an AS External (type 5) LSA and flood the OSPFv2 domain with this routing information. An NSSA can be configured with the following optional parameters:

- No redistribution— Redistributed routes bypass the NSSA and are redistributed to other areas in the OSPFv2 autonomous system. Use this option when the NSSA ASBR is also an ABR.
- Default information originate—Generates an NSSA External (type 7) LSA for a default route to the external autonomous system. Use this option on an NSSA ASBR if the ASBR contains the default route in the routing table. This option can be used on an NSSA ABR whether or not the ABR contains the default route in the routing table.
- Route map—Filters the external routes so that only those routes that you want are flooded throughout the NSSA and other areas.
- Translate—Translates NSSA External LSAs to AS External LSAs for areas outside the NSSA. Use this command on an NSSA ABR to flood the redistributed routes throughout the OSPFv2 autonomous system. You can optionally suppress the forwarding address in these AS External LSAs. If you choose this option, the forwarding address is set to 0.0.0.0.
- No summary—Blocks all summary routes from flooding the NSSA. Use this option on the NSSA ABR.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Ensure that there are no virtual links in the proposed NSSA and that it is not the backbone area.

Procedure

	Command or Action	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**area** *area-id* **nssa [no-redistribution] [default-information-originate]originate** [**route-map** *map-name*]] [**no-summary**] [**translate type7** {**always** | **never**} [**suppress-fa**]] **Example:** `switch(config-router)# area 0.0.0.10 nssa`	Creates this area as an NSSA.
Step 4	(Optional) **area** *area-id* **default-cost** *cost* **Example:** `switch(config-router)# area 0.0.0.10 default-cost 25`	Sets the cost metric for the default summary route sent into this NSSA.
Step 5	(Optional) **show ip ospf** *instance-tag* **Example:** `switch(config-router)# show ip ospf 201`	Displays OSPF information.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Example

This example shows how to create an NSSA that blocks all summary route updates:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# area 0.0.0.10 nssa no-summary
switch(config-router)# copy running-config startup-config
```

This example shows how to create an NSSA that generates a default route:

```
switch# configure terminal
switch(config)# router ospf 201
```

```
switch(config-router)# area 0.0.0.10 nssa default-info-originate
switch(config-router)# copy running-config startup-config
```

This example shows how to create an NSSA that filters external routes and blocks all summary route updates:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# area 0.0.0.10 nssa route-map ExternalFilter no-summary
switch(config-router)# copy running-config startup-config
```

This example shows how to create an NSSA that always translates NSSA External (type 5) LSAs to AS External (type 7) LSAs:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# area 0.0.0.10 nssa translate type 7 always
switch(config-router)# copy running-config startup-config
```

Configuring Virtual Links

A virtual link connects an isolated area to the backbone area through an intermediate area. See the Virtual Links section. You can configure the following optional parameters for a virtual link:

- Authentication—Sets a simple password or MD5 message digest authentication and associated keys.
- Dead interval—Sets the time that a neighbor waits for a Hello packet before declaring the local router as dead and tearing down adjacencies.
- Hello interval—Sets the time between successive Hello packets.
- Retransmit interval—Sets the estimated time between successive LSAs.
- Transmit delay—Sets the estimated time to transmit an LSA to a neighbor.

Note You must configure the virtual link on both routers involved before the link becomes active.

You cannot add a virtual link to a stub area.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Procedure

	Command or Action	Purpose

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

Step 2	**router ospf** *instance-tag* **Example:**	Creates a new OSPFv2 instance with the configured instance tag.

	Command or Action	Purpose
	`switch(config)# router ospf 201` `switch(config-router)#`	

Step 3	**area** *area-id* **virtual link** *router-id*
	Example:
	`switch(config-router)# area 0.0.0.10 virtual-link 10.1.2.3`
	`switch(config-router-vlink)#`
		Creates one end of a virtual link to a remote router. You must create the virtual link on that remote router to complete the link.

Step 4	(Optional) **show ip ospf virtual-link** [**brief**]
	Example:
	`switch(config-router-vlink)# show ip ospf`
	`virtual-link`
		Displays OSPF virtual link information.

Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.
Step 6	(Optional) **authentication** [**key-chain** *key-id* **message-digest** | **null**] **Example:** `switch(config-router-vlink)# authentication message-digest`	Overrides area-based authentication for this virtual link.
Step 7	(Optional) **authentication-key** [**0** | **3**] *key* **Example:** `switch(config-router-vlink)# authentication-key 0 mypass`	Configures a simple password for this virtual link. Use this command if the authentication is not set to key-chain or message-digest. 0 configures the password in clear text. 3 configures the password as 3DES encrypted.
Step 8	(Optional) **dead-interval** *seconds* **Example:** `switch(config-router-vlink)# dead-interval 50`	Configures the OSPFv2 dead interval, in seconds. The range is from 1 to 65535. The default is four times the hello interval, in seconds.
Step 9	(Optional) **hello-interval** *seconds* **Example:** `switch(config-router-vlink)# hello-interval 25`	Configures the OSPFv2 hello interval, in seconds. The range is from 1 to 65535. The default is 10 seconds.
Step 10	(Optional) **message-digest-key** *key-id* **md5** [**0** | **3**] *key* **Example:** `switch(config-router-vlink)# message-digest-key 21 md5 0 mypass`	Configures message digest authentication for this virtual link. Use this command if the authentication is set to message-digest. 0 configures the password in clear text. 3 configures the pass key as 3DES encrypted.

	Command or Action	Purpose
Step 11	(Optional) **retransmit-interval** *seconds* **Example:** `switch(config-router-vlink)# retransmit-interval 50`	Configures the OSPFv2 retransmit interval, in seconds. The range is from 1 to 65535. The default is 5.
Step 12	(Optional) **transmit-delay** *seconds* **Example:** `switch(config-router-vlink)# transmit-delay 2`	Configures the OSPFv2 transmit-delay, in seconds. The range is from 1 to 450. The default is 1.

Example

This example shows how to create a simple virtual link between two ABRs.

The configuration for ABR 1 (router ID 27.0.0.55) is as follows:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# area 0.0.0.10 virtual-link 10.1.2.3
switch(config-router-vlink)# copy running-config startup-config
```

The configuration for ABR 2 (Router ID 10.1.2.3) is as follows:

```
switch# configure terminal
switch(config)# router ospf 101
switch(config-router)# area 0.0.0.10 virtual-link 27.0.0.55
switch(config-router-vlink)# copy running-config startup-config
```

Configuring Redistribution

You can redistribute routes learned from other routing protocols into an OSPFv2 autonomous system through the ASBR.

You can configure the following optional parameters for route redistribution in OSPF:

- Default information originate—Generates an AS External (type 5) LSA for a default route to the external autonomous system.

Note Default information originate ignores **match** statements in the optional route map.

- Default metric—Sets all redistributed routes to the same cost metric.

Note If you redistribute static routes, Cisco NX-OS requires the **default-information originate** command to successfully redistribute the default static route.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Create the necessary route maps used for redistribution.

Procedure

	Command or Action	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**redistribute** {**bgp** *id* | **direct** | **eigrp** *id* | **isis** *id* | **ospf** *id* | **rip** *id* | **static**} **route-map** *map-name* **Example:** `switch(config-router)# redistribute bgp route-map FilterExternalBGP`	Redistributes the selected protocol into OSPF through the configured route map. **Note** If you redistribute static routes, Cisco NX-OS requires the **default-information originate** command to successfully redistribute the default static route.
Step 4	**default-information originate** [**always**] [**route-map** *map-name*] **Example:** `switch(config-router)# default-information-originate route-map DefaultRouteFilter`	Creates a default route into this OSPF domain if the default route exists in the RIB. Use the following optional keywords: • **always**—Always generate the default route of 0.0.0. even if the route does not exist in the RIB. • **route-map**—Generate the default route if the route map returns true. **Note** This command ignores **match** statements in the route map.
Step 5	**default-metric** [*cost*] **Example:** `switch(config-router)# default-metric 25`	Sets the cost metric for the redistributed routes. This command does not apply to directly connected routes. Use a route map to set the default metric for directly connected routes.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Example

This example shows how to redistribute the Border Gateway Protocol (BGP) into OSPF:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# redistribute bgp route-map FilterExternalBGP
switch(config-router)# copy running-config startup-config
```

Limiting the Number of Redistributed Routes

Route redistribution can add many routes to the OSPFv2 route table. You can configure a maximum limit to the number of routes accepted from external protocols. OSPFv2 provides the following options to configure redistributed route limits:

- Fixed limit—Logs a message when OSPFv2 reaches the configured maximum. OSPFv2 does not accept any more redistributed routes. You can optionally configure a threshold percentage of the maximum where OSPFv2 logs a warning when that threshold is passed.
- Warning only—Logs a warning only when OSPFv2 reaches the maximum. OSPFv2 continues to accept redistributed routes.
- Withdraw—Starts the timeout period when OSPFv2 reaches the maximum. After the timeout period, OSPFv2 requests all redistributed routes if the current number of redistributed routes is less than the maximum limit. If the current number of redistributed routes is at the maximum limit, OSPFv2 withdraws all redistributed routes. You must clear this condition before OSPFv2 accepts more redistributed routes.
- You can optionally configure the timeout period.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Procedure

	Command or Action	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**redistribute** {**bgp** *id* | **direct** | **eigrp** *id* | **isis** *id* | **ospf** *id* | **rip** *id* | **static**} **route-map** *map-name* **Example:**	Redistributes the selected protocol into OSPF through the configured route map.

	Command or Action	Purpose
	`switch(config-router)# redistribute bgp route-map FilterExternalBGP`	
Step 4	**redistribute maximum-prefix** *max* [*threshold*] [**warning-only** | **withdraw** [*num-retries timeout*]] **Example:** `switch(config-router)# redistribute maximum-prefix 1000 75 warning-only`	Specifies a maximum number of prefixes that OSPFv2 distributes. The range is from 0 to 65536. Optionally specifies the following: • *threshold*—Percentage of maximum prefixes that trigger a warning message. • **warning-only**—Logs a warning message when the maximum number of prefixes is exceeded. • **withdraw**—Withdraws all redistributed routes. Optionally tries to retrieve the redistributed routes. The *num-retries* range is from 1 to 12. The *timeout* range is 60 to 600 seconds. The default is 300 seconds. Use the **clear ip ospf redistribution** command if all routes are withdrawn.
Step 5	(Optional) **show running-config ospf** **Example:** `switch(config-router)# show running-config ospf`	Displays the OSPFv2 configuration.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Example

This example shows how to limit the number of redistributed routes into OSPF:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# redistribute bgp route-map FilterExternalBGP
switch(config-router)# redistribute maximum-prefix 1000 75
```

Configuring Route Summarization

You can configure route summarization for inter-area routes by configuring an address range that is summarized. You can also configure route summarization for external, redistributed routes by configuring a summary address for those routes on an ASBR. See the Route Summarization section.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Procedure

	Command or Action	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**area** *area-id* **range** *ip-prefix/length* [**no-advertise**] [**cost** *cost*] **Example:** `switch(config-router)# area 0.0.0.10 range 10.3.0.0/16`	Creates a summary address on an ABR for a range of addresses and optionally does not advertise this summary address in a Network Summary (type 3) LSA. The *cost* range is from 0 to 16777215.
Step 4	**summary-address** *ip-prefix/length* [**no-advertise** | **tag** *tag*] **Example:** `switch(config-router)# summary-address 10.5.0.0/16 tag 2`	Creates a summary address on an ASBR for a range of addresses and optionally assigns a tag for this summary address that can be used for redistribution with route maps.
Step 5	(Optional) **show ip ospf summary-address** **Example:** `switch(config-router)# show ip ospf summary-address`	Displays information about OSPF summary addresses.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Example

This example shows how to create summary addresses between areas on an ABR:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)#  area 0.0.0.10 range 10.3.0.0/16
switch(config-router)# copy running-config startup-config
```

This example shows how to create summary addresses on an ASBR:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# summary-address 10.5.0.0/16
switch(config-router)# copy running-config startup-config
```

Configuring Stub Route Advertisements

Use stub route advertisements when you want to limit the OSPFv2 traffic through this router for a short time. See theOSPFv2 Stub Router Advertisements section.

Stub route advertisements can be configured with the following optional parameters:

- On startup—Sends stub route advertisements for the specified announce time.
- Wait for BGP—Sends stub router advertisements until BGP converges.

Note You should not save the running configuration of a router when it is configured for a graceful shutdown because the router continues to advertise a maximum metric after it is reloaded.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Procedure

	Command or Action	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**max-metric router-lsa** [**external-lsa** [*max-metric-value*]] [**include-stub**] [**on-startup** {*seconds* | **wait-for bgp** *tag*}] [**summary-lsa** [*max-metric-value*}]] **Example:** `switch(config-router)# max-metric router-lsa`	Configures OSPFv2 stub route advertisements.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Example

This example shows how to enable the stub router advertisements on startup for the default 600 seconds:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# max-metric router-lsa on-startup
switch(config-router)# copy running-config startup-config
```

Modifying the Default Timers

OSPFv2 includes a number of timers that control the behavior of protocol messages and shortest path first (SPF) calculations. OSPFv2 includes the following optional timer parameters:

- LSA arrival time—Sets the minimum interval allowed between LSAs that arrive from a neighbor. LSAs that arrive faster than this time are dropped.
- Pacing LSAs—Sets the interval at which LSAs are collected into a group and refreshed, checksummed, or aged. This timer controls how frequently LSA updates occur and optimizes how many are sent in an LSA update message (see theFlooding and LSA Group Pacing section).
- Throttle LSAs—Sets the rate limits for generating LSAs. This timer controls how frequently LSAs are generated after a topology change occurs.
- Throttle SPF calculation—Controls how frequently the SPF calculation is run.

At the interface level, you can also control the following timers:

- Retransmit interval—Sets the estimated time between successive LSAs
- Transmit delay—Sets the estimated time to transmit an LSA to a neighbor.

See the Configuring Networks in OSPFv2 section for information about the hello interval and dead timer.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Procedure

	Command or Action	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

	Command or Action	Purpose
Step 3	**timers lsa-arrival** *msec* **Example:** `switch(config-router)# timers lsa-arrival 2000`	Sets the LSA arrival time in milliseconds. The range is from 10 to 600000. The default is 1000 milliseconds.
Step 4	**timers lsa-group-pacing** *seconds* **Example:** `switch(config-router)# timers lsa-group-pacing 1800`	Sets the interval in seconds for grouping LSAs. The range is from 1 to 1800. The default is 240 seconds.
Step 5	**timers throttle lsa** *start-time hold-interval max-time* **Example:** `switch(config-router)# timers throttle lsa 3000 6000 6000`	Sets the rate limit in milliseconds for generating LSAs with the following timers: • *start-time*—The range is from 50 to 5000 milliseconds. The default value is 50 milliseconds. • *hold-interrval*—The range is from 50 to 30,000 milliseconds. The default value is 5000 milliseconds. • *max-time*—The range is from 50 to 30,000 milliseconds. The default value is 5000 milliseconds
Step 6	**timers throttle spf** *delay-time hold-time max-wait* **Example:** `switch(config-router)# timers throttle spf 3000 2000 4000`	Sets the SPF best path schedule initial delay time and the minimum hold time in seconds between SPF best path calculations. The range is from 1 to 600000. The default is no delay time and 5000 millisecond hold time.

Step 7	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)`
		Enters interface configuration mode.

Step 8	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 9	**ip ospf hello-interval** *seconds* **Example:** `switch(config-if)# ip ospf hello-interval 30`	Sets the hello interval for this interface. The range is from 1 to 65535. The default is 10.
Step 10	**ip ospf dead-interval** *seconds* **Example:**	Sets the dead interval for this interface. The range is from 1 to 65535.

	Command or Action	Purpose
	`switch(config-if)# ip ospf dead-interval 30`	
Step 11	**ip ospf retransmit-interval** *seconds* **Example:** `switch(config-if)# ip ospf retransmit-interval 30`	Sets the estimated time in seconds between LSAs transmitted from this interface. The range is from 1 to 65535. The default is 5.

Step 12	**ip ospf transmit-delay** *seconds*
	Example:
	`switch(config-if)# ip ospf transmit-delay 450`
	`switch(config-if)#`
		Sets the estimated time in seconds to transmit an LSA to a neighbor. The range is from 1 to 450. The default is 1.

Step 13	(Optional) **show ip ospf** **Example:** `switch(config-if)# show ip ospf`	Displays information about OSPF.
Step 14	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Example

This example shows how to control LSA flooding with the lsa-group-pacing option:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# timers lsa-group-pacing 300
switch(config-router)# copy running-config startup-config
```

Restarting an OSPFv2 Instance

You can restart an OSPv2 instance. This action clears all neighbors for the instance.

To restart an OSPFv2 instance and remove all associated neighbors, use the following command:

Procedure

	Command or Action	Purpose
Step 1	**restart ospf** *instance-tag* **Example:** `switch(config)# restart ospf 201`	Restarts the OSPFv2 instance and removes all neighbors.

Configuring OSPFv2 with Virtualization

You can create multiple OSPFv2 instances. You can also create multiple VRFs and use the same or multiple OSPFv2 instances in each VRF. You can assign an OSPFv2 interface to a VRF.

Note Configure all other parameters for an interface after you configure the VRF for an interface. Configuring a VRF for an interface deletes all the configuration for that interface.

Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

Procedure

	Command or Action	Purpose

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

Step 2	**vrf context** *vrf-name*
	Example:
	`switch(config)# vrf context RemoteOfficeVRF`
	`switch(config-vrf)#`
		Creates a new VRF and enters VRF configuration mode.

Step 3	**router ospf** *instance-tag*
	Example:
	`switch(config-vrf)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 4	**vrf** *vrf-name*
	Example:
	`switch(config-router)# vrf RemoteOfficeVRF`
	`switch(config-router-vrf)#`
		Enters VRF configuration mode.

Step 5	(Optional) **maximum-paths** *path* **Example:** `switch(config-router-vrf)# maximum-paths 4`	Configures the maximum number of equal OSPFv2 paths to a destination in the route table for this VRF. This feature is used for load balancing.

Step 6	**interface** *interface-type slot/port*
	Example:
	`switch(config-router-vrf)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.

	Command or Action	Purpose
Step 7	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 8	**vrf member** *vrf-name* **Example:** `switch(config-if)# vrf member RemoteOfficeVRF`	Adds this interface to a VRF.
Step 9	**ip address** *ip-prefix/length* **Example:** `switch(config-if)# ip address 192.0.2.1/16`	Configures an IP address for this interface. You must do this step after you assign this interface to a VRF.
Step 10	**ip router ospf** *instance-tag* **area** *area-id* **Example:** `switch(config-if)# ip router ospf 201 area 0`	Assigns this interface to the OSPFv2 instance and area configured.
Step 11	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Example

This example shows how to create a VRF and add an interface to the VRF:

```
switch# configure terminal
switch(config)# vrf context NewVRF
switch(config)# router ospf 201
switch(config)# interface ethernet 1/2
switch(config-if)# vrf member NewVRF
switch(config-if)# ip address 192.0.2.1/16
switch(config-if)# ip router ospf 201 area 0
switch(config-if)# copy running-config startup-config
```

Verifying the OSPFv2 Configuration

To display the OSPFv2 configuration, perform one of the following tasks:

Command	Purpose
show ip ospf	Displays the OSPFv2 configuration.

Command	Purpose
show ip ospf border-routers [**vrf** {*vrf-name* **all** | **default** | **management**}]	Displays the OSPFv2 border router configuration.
show ip ospf database [**vrf** {*vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv2 link-state database summary.
show ip ospf interface *number* [**vrf** {*vrf-name* **all** | **default** | **management**}]	Displays the OSPFv2 interface configuration.
show ip ospf lsa-content-changed-list *neighbor-id interface-type number* [**vrf** {*vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv2 LSAs that have changed.
show ip ospf neighbors [*neighbor-id*] [**detail**] [*interface-type number*] [**vrf** {*vrf-name* **all** | **default** | **management**}] [**summary**]	Displays the list of OSPFv2 neighbors.
show ip ospf request-list *neighbor-id interface-type number* [**vrf** {*vrf-name* **all** | **default** |**management**}]	Displays the list of OSPFv2 link-state requests.
show ip ospf retransmission-list *neighbor-id interface-type number* [**vrf** {*vrf-name* **all** | **default** | **management**}]	Displays the list of OSPFv2 link-state retransmissions.
show ip ospf route [*ospf-route*] [**summary**] [**vrf** {*vrf-name* **all** | **default** | **management**}]	Displays the internal OSPFv2 routes.
show ip ospf summary-address [**vrf** {*vrf-name* **all** | **default** | **management**}]	Displays information about the OSPFv2 summary addresses.
show ip ospf virtual-links [**brief**] [**vrf** {*vrf-name* **all** | **default** | **management**}]	Displays information about OSPFv2 virtual links.
show ip ospf vrf {*vrf-name* **all** | **default** | **management**}	Displays information about VRF-based OSPFv2 configuration.
show running-configuration ospf	Displays the current running OSPFv2 configuration.

Displaying OSPFv2 Statistics

To display OSPFv2 statistics, use the following commands:

Command	Purpose
show ip ospf policy statistics area *area-id* **filter-list** { **in** | **out** } [**vrf** {*vrf-name* | **all** | **default** | **management** }]	Displays the OSPFv2 route policy statistics for an area.
show ip ospf policy statistics redistribute { **bgp** *id* | **direct** | **eigrp** *id* | **ospf** *id* | **rip** *id* | **static** } **vrf** { *vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv2 route policy statistics.

Command	Purpose
show ip ospf statistics [**vrf** { *vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv2 event counters.
show ip ospf traffic [*interface - type number*] [**vrf** { *vrf-name* | **all** | **default** | **management** }]	Displays the OSPFv2 packet counters.

Configuration Examples for OSPFv2

This example shows how to configure OSPFv2:

```
feature ospf
router ospf 201
router-id 290.0.2.1

interface ethernet 1/2
no switchport
ip router ospf 201 area 0.0.0.10
ip ospf authentication
ip ospf authentication-key 0 mypass
```

Additional References

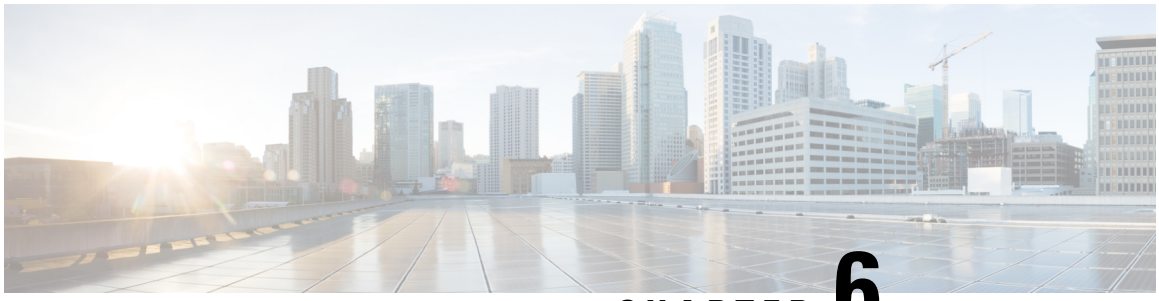
For additional information related to implementing OSPF, see the following sections:

Related Documents

Related Topic	Document Title
Route maps	Configuring Route Policy Manager

MIBs

MIBs	MIBs Link
• OSPF-MIB • OSPF-TRAP-MIB	To locate and download MIBs, go to the following: MIB Locator.



CHAPTER 6

Configuring OSPFv3

This chapter describes how to configure Open Shortest Path First version 3 (OSPFv3) for IPv6 networks on the Cisco NX-OS device.

This chapter includes the following sections:

- Information About OSPFv3, on page 97
- Prerequisites for OSPFv3, on page 108
- Guidelines and Limitations for OSPFv3, on page 109
- Default Settings, on page 109
- Configuring Basic OSPFv3, on page 110
- Configuring Advanced OSPFv3, on page 115
- Verifying the OSPFv3 Configuration, on page 134
- Monitoring OSPFv3, on page 135
- Configuration Examples for OSPFv3, on page 135
- Related Topics, on page 135
- Additional References, on page 136

Information About OSPFv3

OSPFv3 is an IETF link-state protocol (see Overview). An OSPFv3 router sends a special message, called a hello packet, out each OSPF-enabled interface to discover other OSPFv3 neighbor routers. Once a neighbor is discovered, the two routers compare information in the Hello packet to determine if the routers have compatible configurations. The neighbor routers attempt to establish adjacency, which means that the routers synchronize their link-state databases to ensure that they have identical OSPFv3 routing information. Adjacent routers share link-state advertisements (LSAs) that include information about the operational state of each link, the cost of the link, and any other neighbor information. The routers then flood these received LSAs out every OSPF-enabled interface so that all OSPFv3 routers eventually have identical link-state databases. When all OSPFv3 routers have identical link-state databases, the network is converged (see the Convergence section). Each router then uses Dijkstra's Shortest Path First (SPF) algorithm to build its route table.

You can divide OSPFv3 networks into areas. Routers send most LSAs only within one area, which reduces the CPU and memory requirements for an OSPF-enabled router.

OSPFv3 supports IPv6. For information about OSPF for IPv4, see Configuring OSPFv2.

Comparison of OSPFv3 and OSPFv2

Much of the OSPFv3 protocol is the same as in OSPFv2. OSPFv3 is described in RFC 2740.

The key differences between the OSPFv3 and OSPFv2 protocols are as follows:

- OSPFv3 expands on OSPFv2 to provide support for IPv6 routing prefixes and the larger size of IPv6 addresses.
- LSAs in OSPFv3 are expressed as prefix and prefix length instead of address and mask.
- The router ID and area ID are 32-bit numbers with no relationship to IPv6 addresses.
- OSPFv3 uses link-local IPv6 addresses for neighbor discovery and other features.
- OSPFv3 can use the IPv6 authentication trailer (RFC 6506) or IPSec (RFC 4552) for authentication. However, Cisco NX-OS does not support RFC 6506 and provides only partial support for RFC 4552, beginning with Cisco NX-OS release 7.0(3)I3(1).
- OSPFv3 redefines LSA types.

Hello Packet

OSPFv3 routers periodically send Hello packets on every OSPF-enabled interface. The hello interval determines how frequently the router sends these Hello packets and is configured per interface. OSPFv3 uses Hello packets for the following tasks:

- Neighbor discovery
- Keepalives
- Bidirectional communications
- Designated router election (see the Designated Routers section)

The Hello packet contains information about the originating OSPFv3 interface and router, including the assigned OSPFv3 cost of the link, the hello interval, and optional capabilities of the originating router. An OSPFv3 interface that receives these Hello packets determines if the settings are compatible with the receiving interface settings. Compatible interfaces are considered neighbors and are added to the neighbor table (see the Neighbors section).

Hello packets also include a list of router IDs for the routers that the originating interface has communicated with. If the receiving interface sees its own router ID in this list, then bidirectional communication has been established between the two interfaces.

OSPFv3 uses Hello packets as a keepalive message to determine if a neighbor is still communicating. If a router does not receive a Hello packet by the configured dead interval (usually a multiple of the hello interval), then the neighbor is removed from the local neighbor table.

Neighbors

An OSPFv3 interface must have a compatible configuration with a remote interface before the two can be considered neighbors. The two OSPFv3 interfaces must match the following criteria:

- Hello interval

- Dead interval
- Area ID (see the Areas section)
- Authentication
- Optional capabilities

If there is a match, the information is entered into the neighbor table:

- Neighbor ID—The router ID of the neighbor router.
- Priority—Priority of the neighbor router. The priority is used for designated router election (see the Designated Routers section).
- State—Indication of whether the neighbor has just been heard from, is in the process of setting up bidirectional communications, is sharing the link-state information, or has achieved full adjacency.
- Dead time—Indication of how long since the last Hello packet was received from this neighbor.
- Link-local IPv6 Address—The link-local IPv6 address of the neighbor.
- Designated Router—Indication of whether the neighbor has been declared the designated router or backup designated router (see the Designated Routers section).
- Local interface—The local interface that received the Hello packet for this neighbor.

When the first Hello packet is received from a new neighbor, the neighbor is entered into the neighbor table in the initialization state. Once bidirectional communication is established, the neighbor state becomes two-way. ExStart and exchange states come next, as the two interfaces exchange their link-state database. Once this is all complete, the neighbor moves into the full state, which signifies full adjacency. If the neighbor fails to send any Hello packets in the dead interval, then the neighbor is moved to the down state and is no longer considered adjacent.

Adjacency

Not all neighbors establish adjacency. Depending on the network type and designated router establishment, some neighbors become fully adjacent and share LSAs with all their neighbors, while other neighbors do not. For more information, see the Designated Routers section.

Adjacency is established using Database Description packets, Link State Request packets, and Link State Update packets in OSPFv3. The Database Description packet includes the LSA headers from the link-state database of the neighbor (see the Link-State Database section). The local router compares these headers with its own link-state database and determines which LSAs are new or updated. The local router sends a Link State Request packet for each LSA that it needs new or updated information on. The neighbor responds with a Link State Update packet. This exchange continues until both routers have the same link-state information.

Designated Routers

Networks with multiple routers present a unique situation for OSPFv3. If every router floods the network with LSAs, the same link-state information is sent from multiple sources. Depending on the type of network, OSPFv3 might use a single router, the designated router (DR), to control the LSA floods and represent the network to the rest of the OSPFv3 area (see the Areas section). If the DR fails, OSPFv3 selects a backup designated router (BDR). If the DR fails, OSPFv3 uses the BDR.

Network types are as follows:

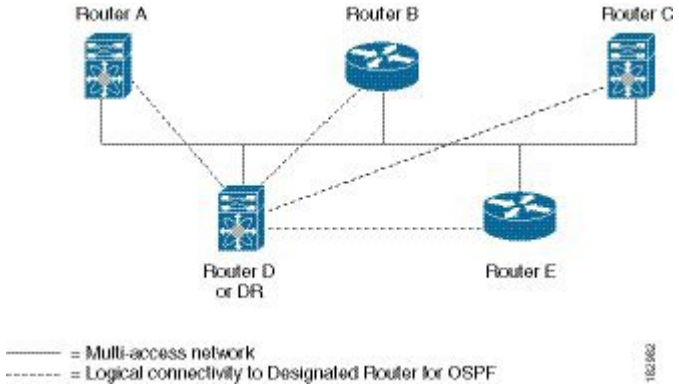
- Point-to-point—A network that exists only between two routers. All neighbors on a point-to-point network establish adjacency and there is no DR.
- Broadcast—A network with multiple routers that can communicate over a shared medium that allows broadcast traffic, such as Ethernet. OSPFv3 routers establish a DR and BDR that controls LSA flooding on the network. OSPFv3 uses the well-known IPv6 multicast addresses, FF02::5, and a MAC address of 0100.5300.0005 to communicate with neighbors.

The DR and BDR are selected based on the information in the Hello packet. When an interface sends a Hello packet, it sets the priority field and the DR and BDR field if it knows who the DR and BDR are. The routers follow an election procedure based on which routers declare themselves in the DR and BDR fields and the priority field in the Hello packet. As a final determinant, OSPFv3 chooses the highest router IDs as the DR and BDR.

All other routers establish adjacency with the DR and the BDR and use the IPv6 multicast address FF02::6 to send LSA updates to the DR and BDR. The following figure shows this adjacency relationship between all routers and the DR.

DRs are based on a router interface. A router might be the DR for one network and not for another network on a different interface

Figure 21: DR in Multi-Access Network



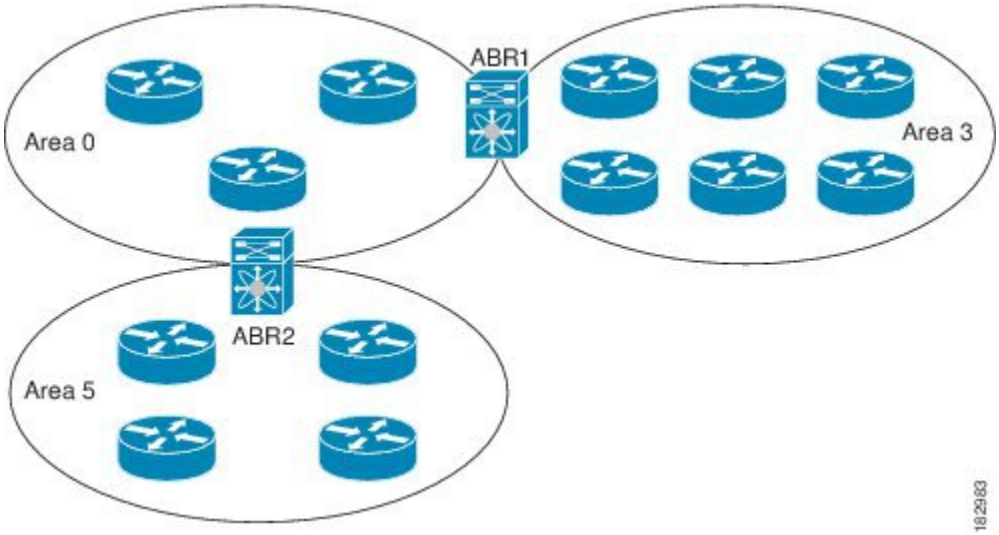
Areas

You can limit the CPU and memory requirements that OSPFv3 puts on the routers by dividing an OSPFv3 network into areas. An area is a logical division of routers and links within an OSPFv3 domain that creates separate subdomains. LSA flooding is contained within an area, and the link-state database is limited to links within the area. You can assign an area ID to the interfaces within the defined area. The Area ID is a 32-bit value that can be expressed as a number or in dotted decimal notation, such as 10.2.3.1.

Cisco NX-OS always displays the area in dotted decimal notation.

If you define more than one area in an OSPFv3 network, you must also define the backbone area, which has the reserved area ID of 0. If you have more than one area, then one or more routers become area border routers (ABRs). An ABR connects to both the backbone area and at least one other defined area (see the following figure).

Figure 22: OSPFv3 Areas



The ABR has a separate link-state database for each area which it connects to. The ABR sends Inter-Area Prefix (type 3) LSAs (see the Route Summarization section) from one connected area to the backbone area. The backbone area sends summarized information about one area to another area. In Figure **OSPFv3 Areas**, Area 0 sends summarized information about Area 5 to Area 3.

OSPFv3 defines one other router type: the autonomous system boundary router (ASBR). This router connects an OSPFv3 area to another autonomous system. An autonomous system is a network controlled by a single technical administration entity. OSPFv3 can redistribute its routing information into another autonomous system or receive redistributed routes from another autonomous system. For more information, see the Advanced Features section.

Link-State Advertisement

OSPFv3 uses link-state advertisements (LSAs) to build its routing table.

LSA Types

Following table shows the LSA types supported by Cisco NX-OS.

Table 11: LSA Types

Name	Name	Description
1	Router LSA	LSA sent by every router. This LSA includes the state and cost of all links but does not include prefix information. Router LSAs trigger an SPF recalculation. Router LSAs are Designated Routers, on page 59flooded to the local OSPFv3 area.

Name	Name	Description
2	Network LSA	LSA sent by the DR. This LSA lists all routers in the multi-access network but does not include prefix information. Network LSAs trigger an SPF recalculation. See the Designated Routers section.
3	Inter-Area Prefix LSA	LSA sent by the area border router to an external area for each destination in local area. This LSA includes the link cost from the border router to the local destination. See the Areas section.
4	Inter-Area Router LSA	LSA sent by the area border router to an external area. This LSA advertises the link cost to the ASBR only. See the Areas section.
5	AS External LSA	LSA generated by the ASBR. This LSA includes the link cost to an external autonomous system destination. AS External LSAs are flooded throughout the autonomous system. See the Areas section.
7	Type-7 LSA	LSA generated by the ASBR within an NSSA. This LSA includes the link cost to an external autonomous system destination. Type-7 LSAs are flooded only within the local NSSA. See the Areas section.
8	Link LSA	LSA sent by every router, using a link-local flooding scope (see the Flooding and LSA Group Pacing section. This LSA includes the link-local address and IPv6 prefixes for this link.
9	Intra-Area Prefix LSA	LSA sent by every router. This LSA includes any prefix or link state changes. Intra-Area Prefix LSAs are flooded to the local OSPFv3 area. This LSA does not trigger an SPF recalculation.

Name	Name	Description
11	Grace LSAs	LSA sent by a restarting router, using a link-local flooding scope. This LSA is used for a graceful restart of OSPFv3. See the High Availability and Graceful Restart section.

Link Cost

Each OSPFv3 interface is assigned a link cost. The cost is an arbitrary number. By default, Cisco NX-OS assigns a cost that is the configured reference bandwidth divided by the interface bandwidth. By default, the reference bandwidth is 40 Gb/s. The link cost is carried in the LSA updates for each link.

Flooding and LSA Group Pacing

OSPFv3 floods LSA updates to different sections of the network, depending on the LSA type. OSPFv3 uses the following flooding scopes:

- Link-local—LSA is flooded only on the local link. Used for Link LSAs and Grace LSAs.
- Area-local—LSA is flooded throughout a single OSPF area only. Used for Router LSAs, Network LSAs, Inter-Area-Prefix LSAs, Inter-Area-Router LSAs, and Intra-Area-Prefix LSAs.
- AS scope—LSA is flooded throughout the routing domain. An AS scope is used for AS External LSAs.

LSA flooding guarantees that all routers in the network have identical routing information. LSA flooding depends on the OSPFv3 area configuration (see the Areas section). The LSAs are flooded based on the link-state refresh time (every 30 minutes by default). Each LSA has its own link-state refresh time.

You can control the flooding rate of LSA updates in your network by using the LSA group pacing feature. LSA group pacing can reduce high CPU or buffer utilization. This feature groups LSAs with similar link-state refresh times to allow OSPFv3 to pack multiple LSAs into an OSPFv3 Update message.

By default, LSAs with link-state refresh times within 10 seconds of each other are grouped together. You should lower this value for large link-state databases or raise it for smaller databases to optimize the OSPFv3 load on your network.

Link-State Database

Each router maintains a link-state database for the OSPFv3 network. This database contains all the collected LSAs and includes information on all the routes through the network. OSPFv3 uses this information to calculate the bast path to each destination and populates the routing table with these best paths.

LSAs are removed from the link-state database if no LSA update has been received within a set interval, called the MaxAge. Routers flood a repeat of the LSA every 30 minutes to prevent accurate link-state information from being aged out. Cisco NX-OS supports the LSA grouping feature to prevent all LSAs from refreshing at the same time. For more information, see the Flooding and LSA Group Pacingsection.

Multi-Area Adjacency

OSPFv3 multi-area adjacency allows you to configure a link on the primary interface that is in more than one area. This link becomes the preferred intra-area link in those areas. Multi-area adjacency establishes a point-to-point unnumbered link in an OSPFv3 area that provides a topological path for that area. The primary adjacency uses the link to advertise an unnumbered point-to-point link in the Router LSA for the corresponding area when the neighbor state is full.

The multi-area interface exists as a logical construct over an existing primary interface for OSPF; however, the neighbor state on the primary interface is independent of the multi-area interface. The multi-area interface establishes a neighbor relationship with the corresponding multi-area interface on the neighboring router. See the Configuring Multi-Area Adjacency section for more information.

OSPFv3 and the IPv6 Unicast RIB

OSPFv3 runs the Dijkstra shortest path first algorithm on the link-state database. This algorithm selects the best path to each destination based on the sum of all the link costs for each link in the path. The shortest path for each destination is then put in the OSPFv3 route table. When the OSPFv3 network is converged, this route table feeds into the IPv6 unicast RIB. OSPFv3 communicates with the IPv6 unicast RIB to do the following:

- Add or remove routes
- Handle route redistribution from other protocols
- Provide convergence updates to remove stale OSPFv3 routes and for stub router advertisements (see the Multiple OSPFv3 Instances section.)

OSPFv3 also runs a modified Dijkstra algorithm for fast recalculation for Inter-Area Prefix, Inter-Area Router, AS-External, type-7, and Intra-Area Prefix (type 3, 4, 5, 7, 8) LSA changes

Address Family Support

Cisco NX-OS supports multiple address families, such as unicast IPv6 and multicast IPv6. OSPFv3 features that are specific to an address family are as follows:

- Default routes
- Route summarization
- Route redistribution
- Filter lists for border routers
- SPF optimization

Use the **address-family ipv6 unicast** command to enter the IPv6 unicast address family configuration mode when configuring these features.

Advanced Features

Cisco NX-OS supports advanced OSPFv3 features that enhance the usability and scalability of OSPFv3 in the network.

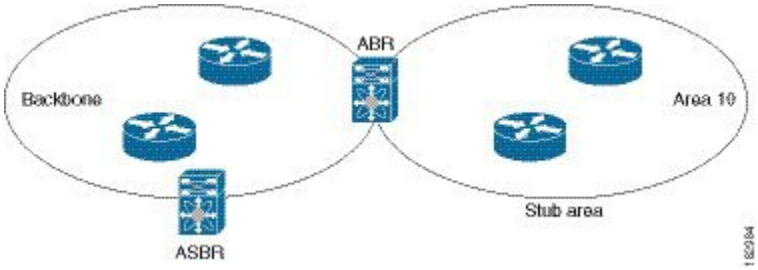
Stub Area

You can limit the amount of external routing information that floods an area by making it a stub area. A stub area is an area that does not allow AS External (type 5) LSAs (see the Link-State Advertisement section). These LSAs are usually flooded throughout the local autonomous system to propagate external route information. Stub areas have the following requirements:

- All routers in the stub area are stub routers. See the Stub Routing section.
- No ASBR routers exist in the stub area.
- You cannot configure virtual links in the stub area.

Following figure shows an example an OSPFv3 autonomous system where all routers in area 0.0.0.10 have to go through the ABR to reach external autonomous systems. Area 0.0.0.10 can be configured as a stub area.

Figure 23: Stub Area



Stub areas use a default route for all traffic that needs to go through the backbone area to the external autonomous system. The default route is an Inter-Area-Prefix LSA with the prefix length set to 0 for IPv6.

Not-So-Stubby Area

A Not-So-Stubby Area (NSSA) is similar to the stub area, except that an NSSA allows you to import autonomous system external routes within an NSSA using redistribution. The NSSA ASBR redistributes these routes and generates type-7 LSAs that it floods throughout the NSSA. You can optionally configure the ABR that connects the NSSA to other areas to translate this type-7 LSA to AS External (type 5) LSAs. The ABR then floods these AS External LSAs throughout the OSPFv3 autonomous system. Summarization and filtering are supported during the translation. See the Link-State Advertisement section for details on type-7 LSAs.

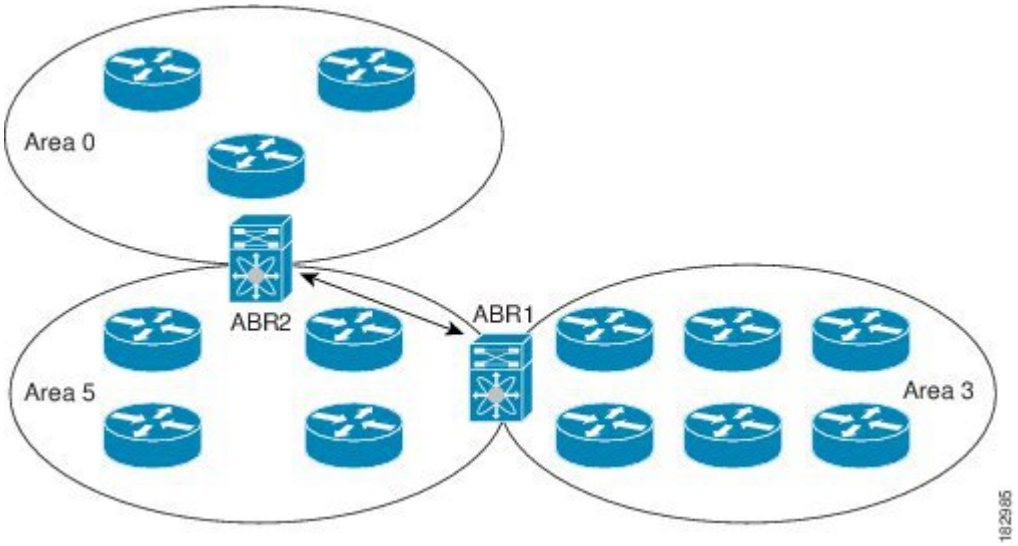
You can, for example, use NSSA to simplify administration if you are connecting a central site using OSPFv3 to a remote site that is using a different routing protocol. Before NSSA, the connection between the corporate site border router and a remote router could not be run as an OSPFv3 stub area because routes for the remote site could not be redistributed into a stub area. With NSSA, you can extend OSPFv3 to cover the remote connection by defining the area between the corporate router and remote router as an NSSA (see the Configuring NSSA, on page 118 section).

The backbone Area 0 cannot be an NSSA.

Virtual Links

Virtual links allow you to connect an OSPFv3 area ABR to a backbone area ABR when a direct physical connection is not available. Following figure shows a virtual link that connects Area 3 to the backbone area through Area 5.

Figure 24: Virtual Links



You can also use virtual links to temporarily recover from a partitioned area, which occurs when a link within the area fails, isolating part of the area from reaching the designated ABR to the backbone area.

Route Redistribution

OSPFv3 can learn routes from other routing protocols by using route redistribution. See the Route Redistribution section. You configure OSPFv3 to assign a link cost for these redistributed routes or a default link cost for all redistributed routes.

Route redistribution uses route maps to control which external routes are redistributed. You must configure a route map with the redistribution to control which routes are passed into OSPFv2. A route map allows you to filter routes based on attributes such as the destination, origination protocol, route type, route tag, and so on. You can use route maps to modify parameters in the AS External (type 5) and NSSA External (type 7) LSAs before these external routes are advertised in the local OSPFv3 autonomous system. For more information, see Configuring Route Policy Manager.

Route Summarization

Because OSPFv3 shares all learned routes with every OSPF-enabled router, you might want to use route summarization to reduce the number of unique routes that are flooded to every OSPF-enabled router. Route summarization simplifies route tables by replacing more-specific addresses with an address that represents all the specific addresses. For example, you can replace 2010:11:22:0:1000::1 and 2010:11:22:0:2000:679:1 with one summary address, 2010:11:22::/32.

Typically, you would summarize at the boundaries of area border routers (ABRs). Although you could configure summarization between any two areas, it is better to summarize in the direction of the backbone so that the backbone receives all the aggregate addresses and injects them, already summarized, into other areas. The two types of summarization are as follows:

- Inter-area route summarization
- External route summarization

You configure inter-area route summarization on ABRs, summarizing routes between areas in the autonomous system. To take advantage of summarization, assign network numbers in areas in a contiguous way to be able to lump these addresses into one range.

External route summarization is specific to external routes that are injected into OSPFv3 using route redistribution. You should make sure that external ranges that are being summarized are contiguous. Summarizing overlapping ranges from two different routers could cause packets to be sent to the wrong destination. Configure external route summarization on ASBRs that are redistributing routes into OSPF.

When you configure a summary address, Cisco NX-OS automatically configures a discard route for the summary address to prevent routing black holes and route loops.

High Availability and Graceful Restart

Cisco NX-OS supports high-availability. If a Cisco NX-OS system experiences a cold reboot, the network stops forwarding traffic to the system and removes the system from the network topology. In this scenario, OSPFv3 experiences a stateless restart, and removes all neighbor adjacencies on the local system. Cisco NX-OS applies the startup configuration and OSPFv3 rediscovers the neighbors and establishes the adjacencies again.

OSPFv3 automatically restarts if the process experiences problems. After the restart, OSPFv3 initiates a graceful restart so that the platform is not taken out of the network topology. If you manually restart OSPF, it performs a graceful restart, which is similar to a stateful switchover. The running configuration is applied in both cases.

A graceful restart, or nonstop forwarding (NSF), allows OSPFv3 to remain in the data forwarding path through a process restart. When OSPFv3 needs to restart, it first sends a link-local Grace (type 11) LSA. This restarting OSPFv3 platform is called NSF capable.

The Grace LSA includes a grace period, which is a specified time that the neighbor OSPFv3 interfaces hold onto the LSAs from the restarting OSPFv3 interface. (Typically, OSPFv3 tears down the adjacency and discards all LSAs from a down or restarting OSPFv3 interface.) The participating neighbors, which are called NSF helpers, keep all LSAs that originate from the restarting OSPFv3 interface as if the interface were still adjacent.

When the restarting OSPFv3 interface is operational again, it rediscovers its neighbors, establishes adjacency, and starts sending its LSA updates again. At this point, the NSF helpers recognize that graceful restart has finished.

Note If the restarting OSPFv3 interface does not come back up before the end of the grace period, or if the network experiences a topology change, the OSPFv3 neighbors tear down adjacency with the restarting OSPFv3 and treat it as a normal OSPFv3 restart.

Note You must enable graceful restart to support an in-service software upgrade (ISSU) for OSPFv3. If you disable graceful restart, Cisco NX-OS issues a warning that ISSU cannot be supported with this configuration.

Multiple OSPFv3 Instances

Cisco NX-OS supports multiple instances of the OSPFv3 protocol. By default, every instance uses the same system router ID. You must manually configure the router ID for each instance if the instances are in the same OSPFv3 autonomous system.

The OSPFv3 header includes an instance ID field to identify that OSPFv3 packet for a particular OSPFv3 instance. You can assign the OSPFv3 instance. The interface drops all OSPFv3 packets that do not have a matching OSPFv3 instance ID in the packet header.

Cisco NX-OS allows only one OSPFv3 instance on an interface.

SPF Optimization

Cisco NX-OS optimizes the SPF algorithm in the following ways:

- Partial SPF for Network (type 2) LSAs, Inter-Area Prefix (type 3) LSAs, and AS External (type 5) LSAs—When there is a change on any of these LSAs, Cisco NX-OS performs a faster partial calculation rather than running the whole SPF calculation.
- SPF timers—You can configure different timers for controlling SPF calculations. These timers include exponential backoff for subsequent SPF calculations. The exponential backoff limits the CPU load of multiple SPF calculations.

BFD

This feature supports bidirectional forwarding detection (BFD). BFD is a detection protocol designed to provide fast forwarding-path failure detection times. BFD provides subsecond failure detection between two adjacent devices and can be less CPU-intensive than protocol hello messages because some of the BFD load can be distributed onto the data plane on supported modules.

Virtualization Support

OSPFv3 supports virtual routing and forwarding (VRF) instances.

Prerequisites for OSPFv3

OSPFv3 has the following prerequisites:

- You must be familiar with routing fundamentals to configure OSPFv3.
- You must be logged on to the switch.
- You have configured at least one interface for IPv6 that is capable of communicating with a remote OSPFv3 neighbor.
- You have installed the Enterprise Services license.
- You have completed the OSPFv3 network strategy and planning for your network. For example, you must decide whether multiple areas are required.
- You have enabled OSPF (see the Enabling OSPFv3, on page 110 section).
- You have installed the Advanced Services license.

- You are familiar with IPv6 addressing and basic configuration. See Configuring IPv6, on page 35 for information on IPv6 routing and addressing.

Guidelines and Limitations for OSPFv3

OSPFv3 has the following configuration guidelines and limitations:

- OSPFv3 IPSEC authentication is not supported.
- Cisco NX-OS displays areas in dotted decimal notation regardless of whether you enter the area in decimal or dotted decimal notation.
- If you configure OSPFv3 in a virtual port channel (vPC) environment, use the following timer commands in router configuration mode on the core switch to ensure fast OSPF convergence when a vPC peer link is shut down:

```
switch (config-router)# timers throttle spf 1 50 50
switch (config-router)# timers lsa-arrival 10
```

Default Settings

Following table lists the default settings for OSPFv3 parameters:

Table 12: Default OSPFv3 Parameters

Parameters	Default
Hello interval	10 seconds
Dead interval	40 seconds
Graceful restart grace period	60 seconds
Graceful restart notify period	15 seconds
OSPFv3 feature	Disabled
Stub router advertisement announce time	600 seconds
Reference bandwidth for link cost calculation	40 Gb/s
LSA minimal arrival time	1000 milliseconds
LSA group pacing	10 seconds
SPF calculation initial delay time	0 milliseconds
SPF calculation hold time	5000 milliseconds
SPF calculation initial delay time	0 milliseconds

Configuring Basic OSPFv3

Configure OSPFv3 after you have designed your OSPFv3 network.

Enabling OSPFv3

You must enable OSPFv3 before you can configure OSPFv3.

Procedure

	Command or Action	Purpose

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

Step 2	**feature ospfv3** **Example:** `switch(config)# feature ospfv3`	Enables OSPFv3.
Step 3	(Optional) **show feature** **Example:** `switch(config)# show feature`	Displays enabled and disabled features.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

To disable the OSPFv3 feature and remove all associated configuration, use the following command in configuration mode.

Command	Purpose
no feature ospfv3 Example: `switch(config)# no feature ospfv`	Disables the OSPFv3 feature and removes all associated configuration.

Creating an OSPFv3 Instance

The first step in configuring OSPFv3 is to create an instance or OSPFv3 instance. You assign a unique instance tag for this OSPFv3 instance. The instance tag can be any string. For each OSPFv3 instance, you can also configure the following optional parameters:

- Router ID—Configures the router ID for this OSPFv3 instance. If you do not use this parameter, the router ID selection algorithm is used. For more information, see the Router IDs section.
- Administrative distance—Rates the trustworthiness of a routing information source. For more information, see the Administrative Distance section.
- Log adjacency changes—Creates a system message whenever an OSPFv3 neighbor changes its state.
- Maximum paths—Sets the maximum number of equal paths that OSPFv3 installs in the route table for a particular destination. Use this parameter for load balancing between multiple paths.
- Reference bandwidth—Controls the calculated OSPFv3 cost metric for a network. The calculated cost is the reference bandwidth divided by the interface bandwidth. You can override the calculated cost by assigning a link cost when a network is added to the OSPFv3 instance. For more information, see the Configuring Networks in OSPFv3 section.

For more information about OSPFv3 instance parameters, see the Configuring Advanced OSPFv3 section.

Before you begin

You must enable OSPFv3 (see the Enabling OSPFv3, on page 110 section).

Ensure that the OSPFv3 instance tag that you plan on using is not already in use on this router.

Use the show **ospfv3** *instance-tag* command to verify that the instance tag is not in use.

OSPFv3 must be able to obtain a router identifier (for example, a configured loopback address) or you must configure the router ID option.

Procedure

	Command or Action	Purpose

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

Step 2	**router ospfv3** *instance-tag*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	(Optional) **router-id** *ip-address* **Example:** `switch(config-router)# router-id 192.0.2.1`	Configures the OSPFv3 router ID. This ID uses the dotted decimal notation and identifies this OSPFv3 instance and must exist on a configured interface in the system.

	Command or Action	Purpose
Step 4	(Optional) **show ipv6 ospfv3** *instance-tag* **Example:** `switch(config-router)# show ipv6 ospfv3 201`	Displays OSPFv3 information.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

To remove the OSPFv3 instance and all associated configuration, use the following command in configuration mode:

Command	Purpose
no router ospfv3 *instance-tag* Example: `switch(config)# no router ospfv3 201`	Deletes the OSPFv3 instance and all associated configuration.

Note This command does not remove OSPF configuration in interface mode. You must manually remove any OSPFv3 commands configured in interface mode.

You can configure the following optional parameters for OSPFv3 in router configuration mode:

Command	Purpose
log-adjacency-changes [**detail**] Example: `switch(config-router)# log-adjacency-changes`	Generates a system message whenever a neighbor changes state.
passive-interface default Example: `switch(config-router)# passive-interface default`	Suppresses routing updates on all interfaces. This command is overridden by the VRF or interface command mode configuration.

You can configure the following optional parameters for OSPFv3 in address family configuration mode:

Command	Purpose
distance *number* Example: `switch(config-router-af)# distance 25`	Configures the administrative distance for this OSPFv3 instance. The range is from 1 to 255. The default is 110.
maximum-paths *paths* Example: `switch(config-router-af)# maximum-paths 4`	Suppresses routing updates on all interfaces. This command is overridden by the VRF or interface command mode configuration.

This example shows how to create an OSPFv3 instance:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# copy running-config startup-config
```

Configuring Networks in OSPFv3

You can configure a network to OSPFv3 by associating it through the interface that the router uses to connect to that network (see the Neighbors section). You can add all networks to the default backbone area (Area 0), or you can create new areas using any decimal number or an IP address.

Note All areas must connect to the backbone area either directly or through a virtual link.

Note OSPFv3 is not enabled on an interface until you configure a valid IPv6 address for that interface.

Before you begin

You must enable OSPFv3 (see the Enabling OSPFv3, on page 110 section).

Procedure

	Command or Action	Purpose

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

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

	Command or Action	Purpose
Step 3	**ipv6 address** *ipv6-prefix/length* **Example:** `switch(config-if)# ipv6 address 2001:0DB8::1/48`	Assigns an IPv6 address to this interface.
Step 4	**ipv6 router ospfv3** *instance-tag* **area** *area-id* [**secondaries none**] **Example:** `switch(config-if)# ipv6 router ospfv3 201 area 0`	Adds the interface to the OSPFv3 instance and area.
Step 5	(Optional) **show ipv6 ospfv3** *instance-tag* interface *interface-type slot/port* **Example:** `switch(config-if)# show ipv6 ospfv3 201 interface ethernet 1/2`	Displays OSPFv3 information.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

You can configure the following optional parameters for OSPFv3 in interface configuration mode:

Command	Purpose
ospfv3 cost *number* Example: `switch(config-if)# ospfv3 cost 25`	Configures the OSPFv3 cost metric for this interface. The default is to calculate a cost metric, based on the reference bandwidth and interface bandwidth. The range is from 1 to 65535.
ospfv3 dead-interval *seconds* Example: `switch(config-if)# ospfv3 dead-interval 50`	Configures the OSPFv3 dead interval, in seconds. The range is from 1 to 65535. The default is four times the hello interval, in seconds.
ospfv3 hello-interval *seconds* Example: `switch(config-if)# ospfv3 hello-interval 25`	Configures the OSPFv3 hello interval, in seconds. The range is from 1 to 65535. The default is 10 seconds.
ospfv3 instance *instance* Example: `switch(config-if)# ospfv3 instance 25`	Configures the OSPFv3 instance ID. The range is from 0 to 255. The default is 0. The instance ID is link-local in scope.

Command	Purpose
ospfv3 mtu-ignore Example: `switch(config-if)# ospfv3 mtu-ignore`	Configures OSPFv3 to ignore any IP maximum transmission unit (MTU) mismatch with a neighbor. The default is to not establish adjacency if the neighbor MTU does not match the local interface MTU.
ospfv3 network { **broadcast** | **point-point** } Example: `switch(config-if)# ospfv3 network broadcast`	Sets the OSPFv3 network type.
[**default** | **no**] **ospfv3 passive-interface** Example: `switch(config-if)# ospfv3 passive-interface`	Suppresses routing updates on the interface. This command overrides the router or VRF command mode configuration. The **default** option removes this interface mode command and reverts to the router or VRF configuration, if present.
ospfv3 priority *number* Example: `switch(config-if)# ospfv3 priority 25`	Configures the OSPFv3 priority, used to determine the DR for an area. The range is from 0 to 255. The default is 1. See the Designated Routers section.
ospfv3 shutdown Example: `switch(config-if)# ospfv3 shutdown`	Shuts down the OSPFv3 instance on this interface.

This example shows how to add a network area 0.0.0.10 in OSPFv3 instance 201:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ipv6 address 2001:0DB8::1/48
switch(config-if)# ipv6 ospfv3 201 area 0.0.0.10
switch(config-if)# copy running-config startup-config
```

Configuring Advanced OSPFv3

Configure OSPFv3 after you have designed your OSPFv3 network.

Configuring Filter Lists for Border Routers

You can separate your OSPFv3 domain into a series of areas that contain related networks. All areas must connect to the backbone area through an area border router (ABR). OSPFv3 domains can connect to external domains as well through an autonomous system border router (ASBR). See the Areas section.

ABRs have the following optional configuration parameters:

- Area range—Configures route summarization between areas. For more information, see the Configuring Route Summarization section.

- Filter list—Filters the Inter-Area Prefix (type 3) LSAs on an ABR that are allowed in from an external area.

ASBRs also support filter lists.

Before you begin

Create the route map that the filter list uses to filter IP prefixes in incoming or outgoing Inter-Area Prefix (type 3) LSAs. See Configuring Route Policy Manager.

You must enable OSPFv3 (see the Enabling OSPFv3, on page 110 section).

Procedure

	Command or Action	Purpose

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

Step 2	**router ospfv3** *instance-tag*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**address-family ipv6 unicast**
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		Enters IPv6 unicast address family mode.

Step 4	**area** *area-id* **filter-list route-map** *map-name* { **in** | **out** } **Example:** `switch(config-router-af)# area 0.0.0.10 filter-list route-map FilterLSAs in`	Filters incoming or outgoing Inter-Area Prefix (type 3) LSAs on an ABR.
Step 5	(Optional) **show ipv6 ospfv3 policy statistics area** *id* **filter-list** { **in** | **out** } **Example:** `switch(config-if)# show ipv6 ospfv3 policy statistics area 0.0.0.10 filter-list in`	Displays OSPFv3 policy information.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to enable graceful restart if it has been disabled:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# address-family ipv6 unicast
switch(config-router-af)# area 0.0.0.10 filter-list route-map FilterLSAs in
switch(config-router-af)# copy running-config startup-config
```

Configuring Stub Areas

You can configure a stub area for part of an OSPFv3 domain where external traffic is not necessary. Stub areas block AS External (type 5) LSAs, limiting unnecessary routing to and from selected networks. See the Stub Area section. You can optionally block all summary routes from going into the stub area.

Before you begin

You must enable OSPF (see the Enabling OSPFv3, on page 110 section).

Ensure that there are no virtual links or ASBRs in the proposed stub area.

Procedure

	Command or Action	Purpose

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

Step 2	**router ospfv3** *instance-tag*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**area** *area-id* **stub** **Example:** `switch(config-router)# area 0.0.0.10 stub`	Creates this area as a stub area.

Step 4	(Optional) **address-family ipv6 unicast**
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		Enters IPv6 unicast address family mode.

Step 5	(Optional) **area** *area-id* **default-cost** *cost* **Example:** `switch(config-router-af)# area 0.0.0.10 default-cost 25`	Sets the cost metric for the default summary route sent into this stub area. The range is from 0 to 16777215.

	Command or Action	Purpose
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to create a stub area that blocks all summary route updates:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# area 0.0.0.10 stub no-summary
switch(config-router)# copy running-config startup-config
```

Configuring a Totally Stubby Area

You can create a totally stubby area and prevent all summary route updates from going into the stub area.

To create a totally stubby area, use the following command in router configuration mode:

Procedure

	Command or Action	Purpose
Step 1	**area** *area-id* **stub no-summary** **Example:** `switch(config-router)# area 20 stub no-summary`	Creates this area as a totally stubby area.

Configuring NSSA

You can configure an NSSA for part of an OSPFv3 domain where limited external traffic is required. You can optionally translate this external traffic to an AS External (type 5) LSA and flood the OSPFv3 domain with this routing information. An NSSA can be configured with the following optional parameters:

- No redistribution— Redistributes routes that bypass the NSSA to other areas in the OSPFv3 autonomous system. Use this option when the NSSA ASBR is also an ABR.
- Default information originate—Generates a Type-7 LSA for a default route to the external autonomous system. Use this option on an NSSA ASBR if the ASBR contains the default route in the routing table. This option can be used on an NSSA ABR whether or not the ABR contains the default route in the routing table.
- Route map—Filters the external routes so that only those routes you want are flooded throughout the NSSA and other areas.
- Translate—Translates Type-7 LSAs to AS External (type 5) LSAs for areas outside the NSSA. Use this command on an NSSA ABR to flood the redistributed routes throughout the OSPFv3 autonomous system. You can optionally suppress the forwarding address in these AS External LSAs.

- No summary—Blocks all summary routes from flooding the NSSA. Use this option on the NSSA ABR.

Before you begin

You must enable OSPF (see the Enabling OSPFv3, on page 110 section).

Ensure that there are no virtual links in the proposed NSSA and that it is not the backbone area.

Procedure

	Command or Action	Purpose

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

Step 2	**router ospfv3** *instance-tag*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**area** *area-id* **nssa** [**no-redistribution**] [**default-information-originate**] [**route-map** *map-name*] [**no-summary**] [**translate type7** { **always** | **never** } [**suppress-fa**]] **Example:** `switch(config-router)# area 0.0.0.10 nssa`	Creates this area as an NSSA.

Step 4	(Optional) **address-family ipv6 unicast**
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		Enters IPv6 unicast address family mode.

Step 5	(Optional) **area** *area-id* **default-cost** *cost* **Example:** `switch(config-router-af)# area 0.0.0.10 default-cost 25`	Sets the cost metric for the default summary route sent into this NSSA. The range is from 0 to 16777215.
Step 6	**copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to create an NSSA that blocks all summary route updates:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# area 0.0.0.10 nssa no-summary
switch(config-router)# copy running-config startup-config
```

This example shows how to create an NSSA that generates a default route:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# area 0.0.0.10 nssa default-info-originate
switch(config-router)# copy running-config startup-config
```

This example shows how to create an NSSA that filters external routes and blocks all summary route updates:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# area 0.0.0.10 nssa route-map ExternalFilter no-summary
switch(config-router)# copy running-config startup-config
```

This example shows how to create an NSSA that always translates Type-7 LSAs to AS External (type 5) LSAs:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# area 0.0.0.10 nssa translate type 7 always
switch(config-router)# copy running-config startup-config
```

This example shows how to create an NSSA that blocks all summary route updates:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# area 0.0.0.10 nssa no-summary
switch(config-router)# copy running-config startup-config
```

Configuring Multi-Area Adjacency

You can add more than one area to an existing OSPFv3 interface. The additional logical interfaces support multi-area adjacency.

Before you begin

You must enable OSPFv3 (see the Enabling OSPFv3, on page 110 section).

Ensure that you have configured a primary area for the interface (see the Configuring Networks in OSPFv3 section).

Procedure

	Command or Action	Purpose

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

Step 2	**interface** *interface-type slot/port* **Example:**	Enters interface configuration mode.

	Command or Action	Purpose
	`switch(config)# interface ethernet 1/2` `switch(config-if)#`	
Step 3	**ipv6 router ospfv3 instance-tag multi-area** *area-id* **Example:** `switch(config-if)# ipv6 router ospfv3 201 multi-area 3`	Adds the interface to another area. **Note** Beginning with Cisco NX-OS Release 7.0(3)I5(1), the instance tag argument is optional. If you do not specify an instance the multi-area configuration is applied to the same instance that is configured for the primary area on tha interface.
Step 4	(Optional) **show ipv6 ospfv3** *instance-tag* **interface** *interface-type slot/port* **Example:** `switch(config-if)# show ipv6 ospfv3 201 interface ethernet 1/2`	Displays OSPFv3 information.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to add a second area to an OSPFv3 interface:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ipv6 address 2001:0DB8::1/48
switch(config-if)# ipv6 ospfv3 201 area 0.0.0.10
switch(config-if)# ipv6 ospfv3 201 multi-area 20
switch(config-if)# copy running-config startup-config
```

Configuring Virtual Links

A virtual link connects an isolated area to the backbone area through an intermediate area. See the Virtual Links section. You can configure the following optional parameters for a virtual link:

- Dead interval—Sets the time that a neighbor waits for a Hello packet before declaring the local router as dead and tearing down adjacencies.
- Hello interval—Sets the time between successive Hello packets.
- Retransmit interval—Sets the estimated time between successive LSAs.
- Transmit delay—Sets the estimated time to transmit an LSA to a neighbor.

Note You must configure the virtual link on both routers involved before the link becomes active.

Before you begin

You must enable OSPF (see the Enabling OSPFv3, on page 110 section).

Procedure

	Command or Action	Purpose

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

Step 2	**router ospfv3** *instance-tag*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**area** *area-id* **virtual-link** *router-id*
	Example:
	`switch(config-router)# area 0.0.0.10 virtual-link 2001:0DB8::1`
	`switch(config-router-vlink)#`
		Creates one end of a virtual link to a remote router. You must create the virtual link on that remote router to complete the link.

Step 4	(Optional) **show ipv6 ospfv3 virtual-link** [**brief**] **Example:** `switch(config-if)# show ipv6 ospfv3 virtual-link`	Displays OSPFv3 virtual link information.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	Saves this configuration change.

Example

You can configure the following optional commands in virtual link configuration mode:

Command	Purpose
dead-interval *seconds* Example: `switch(config-router-vlink)# dead-interval 50`	Configures the OSPFv3 dead interval, in seconds. The range is from 1 to 65535. The default is four times the hello interval, in seconds.

Command	Purpose
hello-interval *seconds* Example: `switch(config-router-vlink)# hello-interval 25`	Configures the OSPFv3 hello interval, in seconds. The range is from 1 to 65535. The default is 10 seconds.
retransmit-interval *seconds* Example: `switch(config-router-vlink)# retransmit-interval 50`	Configures the OSPFv3 retransmit interval, in seconds. The range is from 1 to 65535. The default is 5.
transmit-delay *seconds* Example: `switch(config-router-vlink)# transmit-delay 2`	Configures the OSPFv3 transmit-delay, in seconds. The range is from 1 to 450. The default is 1.

These examples show how to create a simple virtual link between two ABRs:

Configuration for ABR 1 (router ID 2001:0DB8::1) is as follows:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# area 0.0.0.10 virtual-link 2001:0DB8::10
switch(config-router)# copy running-config startup-config
```

Configuration for ABR 2 (router ID 2001:0DB8::10) is as follows:

```
switch# configure terminal
switch(config)# router ospf 101
switch(config-router)# area 0.0.0.10 virtual-link 2001:0DB8::1
switch(config-router)# copy running-config startup-config
```

Configuring Redistribution

You can redistribute routes learned from other routing protocols into an OSPFv3 autonomous system through the ASBR.

You can configure the following optional parameters for route redistribution in OSPF:

- Default information originate—Generates an AS External (type 5) LSA for a default route to the external autonomous system.

Note Default information originate ignores **match** statements in the optional route map.

- Default metric—Sets all redistributed routes to the same cost metric.

Note If you redistribute static routes, Cisco NX-OS requires the **default-information originate** command to successfully redistribute the default static route.

Before you begin

Create the necessary route maps used for redistribution.

You must enable OSPF (see the Enabling OSPFv3, on page 110 section).

Procedure

	Command or Action	Purpose

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

Step 2	**router ospfv3** *instance-tag*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**address-family ipv6 unicast**
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		Enters IPv6 unicast address family mode.

Step 4	**redistribute** { **bgp** *id* | **direct** | **isis** *id* | **rip** *id* | **static** } **route-map** *map-name* **Example:** `switch(config-router-af)# redistribute bgp route-map FilterExternalBGP`	Redistributes the selected protocol into OSPFv3 through the configured route map. **Note** If you redistribute static routes, Cisco NX-OS requires the **default-information originate** command to successfully redistribute the default static route.
Step 5	**default-information originate** [**always**] [**route-map** *map-name*] **Example:** `switch(config-router-af)# default-information-originate route-map DefaultRouteFilter`	Creates a default route into this OSPFv3 domain if the default route exists in the RIB. Use the following optional keywords: • **always** —Always generates the default route of 0.0.0. even if the route does not exist in the RIB. • **route-map** —Generates the default route if the route map returns true.

	Command or Action	Purpose
		Note This command ignores **match** statements in the route map
Step 6	**default-metric** *cost* **Example:** `switch(config-router-af)# default-metric 25`	Sets the cost metric for the redistributed routes. The range is from 1 to 16777214. This command does not apply to directly connected routes. Use a route map to set the default metric for directly connected routes.
Step 7	**copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to redistribute the Border Gateway Protocol (BGP) into OSPFv3:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# address-family ipv6 unicast
switch(config-router-af)# redistribute bgp route-map FilterExternalBGP
switch(config-router-af)# copy running-config startup-confi
```

Limiting the Number of Redistributed Routes

Route redistribution can add many routes to the OSPFv3 route table. You can configure a maximum limit to the number of routes accepted from external protocols. OSPFv3 provides the following options to configure redistributed route limits:

- Fixed limit—Logs a message when OSPFv3 reaches the configured maximum. OSPFv3 does not accept any more redistributed routes. You can optionally configure a threshold percentage of the maximum where OSPFv3 logs a warning when that threshold is passed.
- Warning only—Logs a warning only when OSPFv3 reaches the maximum. OSPFv3 continues to accept redistributed routes.
- Withdraw—Starts the configured timeout period when OSPFv3 reaches the maximum. After the timeout period, OSPFv3 requests all redistributed routes if the current number of redistributed routes is less than the maximum limit. If the current number of redistributed routes is at the maximum limit, OSPFv3 withdraws all redistributed routes. You must clear this condition before OSPFv3 accepts more redistributed routes. You can optionally configure the timeout period.

Before you begin

You must enable OSPF (see the Enabling OSPFv3, on page 110 section).

Procedure

	Command or Action	Purpose

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

Step 2	**router ospfv3** *instance-tag*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**address-family ipv6 unicast**
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		Enters IPv6 unicast address family mode.

Step 4	**redistribute** { **bgp** *id* | **direct** | **isis** *id* | **rip** *id* | **static** } **route-map** *map-name* **Example:** `switch(config-router-af)# redistribute bgp route-map FilterExternalBGP`	Redistributes the selected protocol into OSPFv3 through the configured route map.
Step 5	**redistribute maximum-prefix** *max* [*threshold*] [**warning-only** | **withdraw** [*num-retries timemout*]] **Example:** `switch(config-router)# redistribute maximum-prefix 1000 75 warning-only`	Specifies a maximum number of prefixes that OSPFv2 distributes. The range is from 0 to 65536. Optionally, specifies the following: • **threshold** —Percent of maximum prefixes that triggers a warning message. • **warning-only** —Logs an warning message when the maximum number of prefixes is exceeded. • **withdraw** —Withdraws all redistributed routes and optionally tries to retrieve the redistributed routes. The*num-retries* range is from 1 to 12. The *timeout* range is from 60 to 600 seconds. The default is 300 seconds.
Step 6	(Optional) **show running-config ospfv3** **Example:** `switch(config-router)# show running-config ospf`	Displays the OSPFv3 configuration.
Step 7	(Optional) copy running-config startup-config **Example:**	Saves this configuration change.

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

Example

This example shows how to limit the number of redistributed routes into OSPF:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# address-family ipv6 unicast
switch(config-router-af)# redistribute bgp route-map FilterExternalBGP
switch(config-router-af)# redistribute maximum-prefix 1000 75
```

Configuring Route Summarization

You can configure route summarization for inter-area routes by configuring an address range that is summarized. You can also configure route summarization for external, redistributed routes by configuring a summary address for those routes on an ASBR. For more information, see the Route Summarization section.

Before you begin

You must enable OSPF (see the Enabling OSPFv3, on page 110 section).

Procedure

	Command or Action	Purpose

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

Step 2	**router ospfv3** *instance-tag*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**address-family ipv6 unicast**
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		Enters IPv6 unicast address family mode.

Step 4	**area** *area-id* **range** *ipv6-prefix/length* [**no-advertise**] [**cost** *cost*] **Example:** `switch(config-router-af)# area 0.0.0.10 range 2001:0DB8::/48 advertise`	Creates a summary address on an ABR for a range of addresses*and o* ptionally advertises this summary address in a Inter-Area Prefix (type 3) LSA. The *cost* range is from 0 to 16777215.

	Command or Action	Purpose
Step 5	**summary-address** *ipv6-prefix/length* [**no-advertise**] [**tag** *tag*] **Example:** `switch(config-router-af)# summary-address 2001:0DB8::/48 tag 2`	Creates a summary address on an ASBR for a range of addresses and optionally assigns a tag for this summary address that can be used for redistribution with route maps.
Step 6	(Optional) **show ipv6 ospfv3 summary-address** **Example:** `switch(config-router)# show ipv6 ospfv3 summary-addres`	Displays information about OSPFv3 summary addresses
Step 7	(Optional) **copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to create summary addresses between areas on an ABR:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# address-family ipv6 unicast
switch(config-router)# area 0.0.0.10 range 2001:0DB8::/48
switch(config-router)# copy running-config startup-config
```

This example shows how to create summary addresses on an ASBR:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# address-family ipv6 unicast
switch(config-router)# summary-address 2001:0DB8::/48
switch(config-router)# copy running-config startup-config
```

Modifying the Default Timers

OSPFv3 includes a number of timers that control the behavior of protocol messages and shortest path first (SPF) calculations. OSPFv3 includes the following optional timer parameters:

- LSA arrival time—Sets the minimum interval allowed between LSAs arriving from a neighbor. LSAs that arrive faster than this time are dropped.
- Pacing LSAs—Sets the interval at which LSAs are collected into a group and refreshed, checksummed, or aged. This timer controls how frequently LSA updates occur and optimizes how many are sent in an LSA update message (see theFlooding and LSA Group Pacing section).
- Throttle LSAs—Sets rate limits for generating LSAs. This timer controls how frequently LSAs are generated after a topology change occurs.
- Throttle SPF calculation—Controls how frequently the SPF calculation is run.

At the interface level, you can also control the following timers:

- Retransmit interval—Sets the estimated time between successive LSAs.
- Transmit delay—Sets the estimated time to transmit an LSA to a neighbor.

See the Configuring Networks in OSPFv3 section for information on the hello interval and dead timer.

Procedure

	Command or Action	Purpose

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

Step 2	**router ospfv3** *instance-tag*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**timers lsa-arrival** **Example:** `switch(config-router)# timers lsa-arrival 2000`	Sets the LSA arrival time in milliseconds. The range is from 10 to 600000. The default is 1000 milliseconds.
Step 4	**timers lsa-group-pacing** *seconds* **Example:** `switch(config-router)# timers lsa-group-pacing 200`	Sets the interval in seconds for grouping LSAs. The range is from 1 to 1800. The default is 10 seconds.
Step 5	**timers throttle lsa** *start-time hold-interval max-time* **Example:** `switch(config-router)# timers throttle lsa network 350 5000 6000`	Sets the rate limit in milliseconds for generating LSAs. You can configure the following timers: **start-time** —The range is from 50 to 5000 milliseconds. The default value is 50 milliseconds. **hold-interval**—The range is from 50 to 30,000 milliseconds. The default value is 5000 milliseconds. **max-time** —The range is from 50 to 30,000 milliseconds. The default value is 5000 milliseconds.

Step 6	**address-family ipv6 unicast**
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		Enters IPv6 unicast address family mode.

	Command or Action	Purpose
Step 7	**timers throttle spf** *delay-time hold-time* **Example:** `switch(config-router)# timers throttle spf 3000 2000`	Sets the SPF best-path schedule initial delay time and the minimum hold time in seconds between SPF best-path calculations. The range is from 1 to 600000. The default is no delay time and 5000 millisecond hold time.

Step 8	**interface** *interface type slot/port*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 9	**ospfv3 retransmit-interval** *seconds* **Example:** `switch(config-if)# ospfv3 retransmit-interval 30`	Sets the estimated time in seconds between LSAs transmitted from this interface. The range is from 1 to 65535. The default is 5.

Step 10	**ospfv3 transmit-delay** *seconds*
	Example:
	`switch(config-if)# ospfv3 transmit-delay 600`
	`switch(config-if)#`
		Sets the estimated time in seconds to transmit an LSA to a neighbor. The range is from 1 to 450. The default is 1.

Step 11	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to control LSA flooding with the lsa-group-pacing option:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# timers lsa-group-pacing 300
switch(config-router)# copy running-config startup-config
```

Configuring Graceful Restart

Graceful restart is enabled by default. You can configure the following optional parameters for graceful restart in an OSPFv3 instance:

- Grace period—Configures how long neighbors should wait after a graceful restart has started before tearing down adjacencies.
- Helper mode disabled—Disables helper mode on the local OSPFv3 instance. OSPFv3 does not participate in the graceful restart of a neighbor.

- Planned graceful restart only—Configures OSPFv3 to support graceful restart only in the event of a planned restart.

Before you begin

You must enable OSPFv3 (see the Enabling OSPFv3, on page 110 section).

Ensure that all neighbors are configured for graceful restart with matching optional parameters set.

Procedure

	Command or Action	Purpose

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

Step 2	**router ospfv3 instance-tag**
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**graceful-restart** **Example:** `switch(config-router)# graceful-restart`	Enables graceful restart. A graceful restart is enabled by default.
Step 4	**graceful-restart grace-period** *seconds* **Example:** `switch(config-router)# graceful-restart grace-period 120`	Sets the grace period, in seconds. The range is from 5 to 1800. The default is 60 seconds.
Step 5	**graceful-restart helper-disable** **Example:** `switch(config-router)# graceful-restart helper-disable`	Disables helper mode. Enabled by default.
Step 6	**graceful-restart planned-only** **Example:** `switch(config-router)# graceful-restart planned-only`	Configures graceful restart for planned restarts only.
Step 7	(Optional) **show ipv6 ospfv3** *instance-tag* **Example:** `switch(config-if)# show ipv6 ospfv3 201`	Displays OSPFv3 information.
Step 8	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to enable graceful restart if it has been disabled and set the grace period to 120 seconds:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# graceful-restart
switch(config-router)# graceful-restart grace-period 120
switch(config-router)# copy running-config startup-config
```

Restarting an OSPFv3 Instance

You can restart an OSPv3 instance. This action clears all neighbors for the instance.

To restart an OSPFv3 instance and remove all associated neighbors, use the following command:

Procedure

	Command or Action	Purpose
Step 1	**restart ospfv3** *instance-tag* **Example:** `switch(config)# restart ospfv3 201`	Restarts the OSPFv3 instance and removes all neighbors.

Configuring OSPFv3 with Virtualization

You can configure multiple OSPFv3 instances in each VDC. You can also create multiple VRFs within each VDC and use the same or multiple OSPFv3 instances in each VRF. You assign an OSPFv3 interface to a VRF.

Note Configure all other parameters for an interface after you configure the VRF for an interface. Configuring a VRF for an interface deletes all the configuration for that interface.

Before you begin

Create the VDCs.

You must enable OSPF (see the Enabling OSPFv3, on page 110 section).

Procedure

	Command or Action	Purpose

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

	Command or Action	Purpose

Step 2	**vrf context** *vrf-name*
	Example:
	`switch(config)# vrf context RemoteOfficeVRF`
	`switch(config-vrf)#`
		Creates a new VRF and enters VRF configuration mode.

Step 3	**router ospfv3** *instance-tag*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 4	**vrf** *vrf-name*
	Example:
	`switch(config-router)# vrf RemoteOfficeVRF`
	`switch(config-router-vrf)#`
		Enters VRF configuration mode.

Step 5	(Optional) **maximum-paths** *paths* **Example:** `switch(config-router-vrf)# maximum-paths 4`	Configures the maximum number of equal OSPFv3 paths to a destination in the route table for this VRF. Use this command for load balancing.

Step 6	**interface** *interface type slot/port*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 7	**vrf member** *vrf-name* **Example:** `switch(config-if)# vrf member RemoteOfficeVR`	Adds this interface to a VRF.
Step 8	**ipv6 address** *ipv6-prefix/length* **Example:** `switch(config-if)# ipv6 address 2001:0DB8::1/48`	Configures an IP address for this interface. You must do this step after you assign this interface to a VRF.
Step 9	**ipv6 ospfv3** *instance-tag* **area** *area-id* **Example:** `switch(config-if)# ipv6 ospfv3 201 area 0`	Assigns this interface to the OSPFv3 instance and area configured.
Step 10	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to create a VRF and add an interface to the VRF:

```
switch# configure terminal
switch(config)# vrf context NewVRF
switch(config-vrf)# exit
switch(config)# router ospfv3 201
switch(config-router)# exit
switch(config)# interface ethernet 1/2
switch(config-if)# vrf member NewVRF
switch(config-if)# ipv6 address 2001:0DB8::1/48
switch(config-if)# ipv6 ospfv3 201 area 0
switch(config-if)# copy running-config startup-config
```

Verifying the OSPFv3 Configuration

To display the OSPFv3 configuration, perform one of the following tasks:

Command	Purpose
show ipv6 ospfv3	Displays the OSPFv3 configuration.
show ipv6 ospfv3 border-routers	Displays the internal OSPF routing table entries to an ABR and ASBR
show ipv6 ospfv3 database	Displays lists of information related to the OSPFv3 database for a specific router.
show ipv6 ospfv3 interface *type number* [**vrf** { *vrf-name* | **all** | **default** | **management** }]	Displays the OSPFv3 interface configuration.
show ipv6 ospfv3 neighbors	Displays the neighbor information. Use the **clear ospfv3 neighbors** command to remove adjacency with all neighbors.
show ipv6 ospfv3 request-list	Displays a list of LSAs requested by a router.
show ipv6 ospfv3 retransmission-list	Displays a list of LSAs waiting to be retransmitted.
show ipv6 ospfv3 summary-address	Displays a list of all summary address redistribution information configured under an OSPFv3 instance.
show ospfv3 process	Displays the OSPFv3 authentication configuration at the process level.

Command	Purpose
show ospfv3 interface *interface-type slot/port*	Displays the OSPFv3 authentication configuration at the interface level.
show running-configuration ospfv3	Displays the current running OSPFv3 configuration.

Monitoring OSPFv3

To display OSPFv3 statistics, use the following commands:

Command	Purpose
show ipv6 ospfv3 memory	Displays the OSPFv3 memory usage statistics.
show ipv6 ospfv3 policy statistics area *area-id* **filter-list** {**in** | **out**} [**vrf** {*vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv3 route policy statistics for an area.
show ipv6 ospfv3 policy statistics redistribute {**bgp** *id* | **direct** | **isis** *id* | **rip** *id* | **static**} **vrf** {*vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv3 route policy statistics.
show ipv6 ospfv3 statistics [**vrf** {*vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv3 event counters
show ipv6 ospfv3 traffic [*interface-type number*] [**vrf** {*vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv3 packet counters.

Configuration Examples for OSPFv3

This example shows how to configure OSPFv3:

```
feature ospfv3
router ospfv3 201
router-id 290.0.2.1

interface ethernet 1/2
ipv6 address 2001:0DB8::1/48

ipv6 ospfv3 201 area 0.0.0.10
```

Related Topics

The following topics can give more information on OSPF:

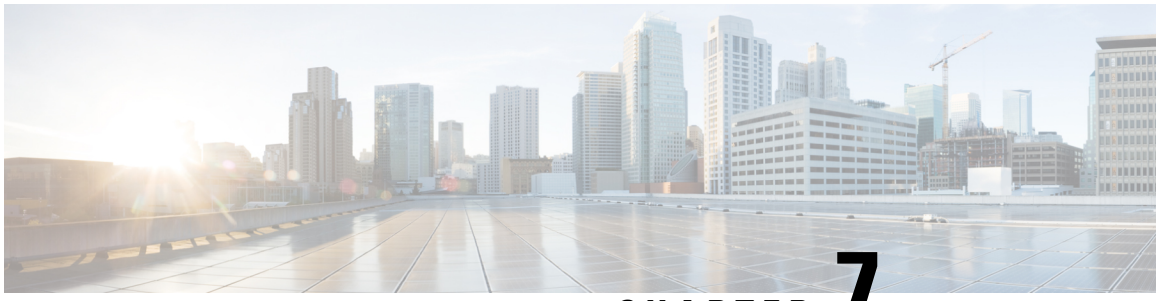
- Configuring OSPFv2
- Configuring Route Policy Manager

Additional References

For additional information related to implementing OSPF, see the following sections:

MIBs

MIBs	MIBs Link
• OSPF-MIB • OSPF-IRAP-MIB	To locate and download MIBs, go to the following: MIB Locator.



CHAPTER 7

Configuring EIGRP

This chapter describes how to configure the Enhanced Interior Gateway Routing Protocol (EIGRP) on Cisco NX-OS switches.

This chapter includes the following sections:

- Information About EIGRP, on page 137
- Prerequisites for EIGRP, on page 143
- Guidelines and Limitations, on page 143
- Default Settings, on page 143
- Configuring Basic EIGRP, on page 144
- Configuring Advanced EIGRP, on page 148
- Configuring Virtualization for EIGRP, on page 159
- Verifying the EIGRP Configuration, on page 161
- Displaying EIGRP Statistics, on page 161
- Configuration Examples for EIGRP, on page 161
- Related Topics, on page 162
- Additional References, on page 162

Information About EIGRP

EIGRP combines the benefits of distance vector protocols with the features of link-state protocols. EIGRP sends out periodic hello messages for neighbor discovery. Once EIGRP learns a new neighbor, it sends a one-time update of all the local EIGRP routes and route metrics. The receiving EIGRP router calculates the route distance based on the received metrics and the locally assigned cost of the link to that neighbor. After this initial full route table update, EIGRP sends incremental updates to only those neighbors affected by the route change. This process speeds convergence and minimizes the bandwidth used by EIGRP.

EIGRP Components

EIGRP has the following basic components:

- Reliable Transport Protocol
- Neighbor Discovery and Recovery
- Diffusing Update Algorithm

Reliable Transport Protocol

The Reliable Transport Protocol guarantees ordered delivery of EIGRP packets to all neighbors. (See the Neighbor Discovery and Recovery section.) The Reliable Transport Protocol supports an intermixed transmission of multicast and unicast packets. The reliable transport can send multicast packets quickly when unacknowledged packets are pending. This provision helps to ensure that the convergence time remains low for various speed links. See the Configuring Advanced EIGRP section for details about modifying the default timers that control the multicast and unicast packet transmissions.

The Reliable Transport Protocol includes the following message types:

- **Hello**—Used for neighbor discovery and recovery. By default, EIGRP sends a periodic multicast hello message on the local network at the configured hello interval. By default, the hello interval is 5 seconds.
- **Acknowledgement**—Verifies reliable reception of Updates, Queries, and Replies.
- **Updates**—Sends to affected neighbors when routing information changes. Updates include the route destination, address mask, and route metrics such as delay and bandwidth. The update information is stored in the EIGRP topology table.
- **Queries and Replies**—Sent as necessary as part of the Diffusing Update Algorithm used by EIGRP.

Neighbor Discovery and Recovery

EIGRP uses the hello messages from the Reliable Transport Protocol to discover neighboring EIGRP routers on directly attached networks. EIGRP adds neighbors to the neighbor table. The information in the neighbor table includes the neighbor address, the interface it was learned on, and the hold time, which indicates how long EIGRP should wait before declaring a neighbor unreachable. By default, the hold time is three times the hello interval or 15 seconds.

EIGRP sends a series of Update messages to new neighbors to share the local EIGRP routing information. This route information is stored in the EIGRP topology table. After this initial transmission of the full EIGRP route information, EIGRP sends Update messages only when a routing change occurs. These Update messages contain only the new or changed information and are sent only to the neighbors affected by the change. See the EIGRP Route Updates section.

EIGRP also uses the hello messages as a keepalive to its neighbors. As long as hello messages are received, Cisco NX-OS can determine that a neighbor is alive and functioning.

Diffusing Update Algorithm

The Diffusing Update Algorithm (DUAL) calculates the routing information based on the destination networks in the topology table. The topology table includes the following information:

- IPv4 address/mask—The network address and network mask for this destination.
- Successors—The IP address and local interface connection for all feasible successors or neighbors that advertise a shorter distance to the destination than the current feasible distance.
- Feasibility distance (FD)—The lowest calculated distance to the destination. The feasibility distance is the sum of the advertised distance from a neighbor plus the cost of the link to that neighbor.

DUAL uses the distance metric to select efficient, loop-free paths. DUAL selects routes to insert into the unicast Routing Information Base (RIB) based on feasible successors. When a topology change occurs, DUAL looks for feasible successors in the topology table. If there are feasible successors, DUAL selects the feasible

successor with the lowest feasible distance and inserts that into the unicast RIB, avoiding unnecessary recomputation.

When there are no feasible successors but there are neighbors advertising the destination, DUAL transitions from the passive state to the active state and triggers a recomputation to determine a new successor or next-hop router to the destination. The amount of time required to recompute the route affects the convergence time. EIGRP sends Query messages to all neighbors, searching for feasible successors. Neighbors that have a feasible successor send a Reply message with that information. Neighbors that do not have feasible successors trigger a DUAL recomputation.

EIGRP Route Updates

When a topology change occurs, EIGRP sends an Update message with only the changed routing information to affected neighbors. This Update message includes the distance information to the new or updated network destination.

The distance information in EIGRP is represented as a composite of available route metrics, including bandwidth, delay, load utilization, and link reliability. Each metric has an associated weight that determines if the metric is included in the distance calculation. You can configure these metric weights. You can fine-tune link characteristics to achieve optimal paths, but we recommend that you use the default settings for most configurable metrics.

Internal Route Metrics

Internal routes are routes that occur between neighbors within the same EIGRP autonomous system. These routes have the following metrics:

- Next hop—The IP address of the next-hop router.
- Delay—The sum of the delays configured on the interfaces that make up the route to the destination network. Configured in tens of microseconds.
- Bandwidth—The calculation from the lowest configured bandwidth on an interface that is part of the route to the destination.

Note We recommend that you use the default bandwidth value. This bandwidth parameter is also used by EIGRP.

- MTU—The smallest maximum transmission unit value along the route to the destination.
- Hop count—The number of hops or routers that the route passes through to the destination. This metric is not directly used in the DUAL computation.
- Reliability—An indication of the reliability of the links to the destination.
- Load—An indication of how much traffic is on the links to the destination.

By default, EIGRP uses the bandwidth and delay metrics to calculate the distance to the destination. You can modify the metric weights to include the other metrics in the calculation.

External Route Metrics

External routes are routes that occur between neighbors in different EIGRP autonomous systems. These routes have the following metrics:

- Next hop—The IP address of the next-hop router.
- Router ID—The router ID of the router that redistributed this route into EIGRP.
- AS Number—The autonomous system number of the destination.
- Protocol ID—A code that represents the routing protocol that learned the destination route.
- Tag—An arbitrary tag that can be used for route maps.
- Metric—The route metric for this route from the external routing protocol.

EIGRP and the Unicast RIB

EIGRP adds all learned routes to the EIGRP topology table and the unicast RIB. When a topology change occurs, EIGRP uses these routes to search for a feasible successor. EIGRP also listens for notifications from the unicast RIB for changes in any routes redistributed to EIGRP from another routing protocol.

Advanced EIGRP

You can use the advanced features of EIGRP to optimize your EIGRP configuration.

Address Families

EIGRP supports the IPv4 address family.

Address family configuration mode includes the following EIGRP features:

- Authentication
- AS number
- Default route
- Metrics
- Distance
- Graceful restart
- Logging
- Load balancing
- Redistribution
- Router ID
- Stub router
- Timers

You cannot configure the same feature in more than one configuration mode. For example, if you configure the default metric in router configuration mode, you cannot configure the default metric in address family mode.

Authentication

You can configure authentication on EIGRP messages to prevent unauthorized or invalid routing updates in your network. EIGRP authentication supports MD5 authentication digest.

You can configure the EIGRP authentication per virtual routing and forwarding (VRF) instance or interface using keychain management for the authentication keys. Keychain management allows you to control changes to the authentication keys used by MD5 authentication digest.

For MD5 authentication, you configure a password that is shared at the local router and all remote EIGRP neighbors. When an EIGRP message is created, Cisco NX-OS creates an MD5 one-way message digest based on the message itself and the encrypted password and sends this digest along with the EIGRP message. The receiving EIGRP neighbor validates the digest using the same encrypted password. If the message has not changed, the calculation is identical and the EIGRP message is considered valid.

MD5 authentication also includes a sequence number with each EIGRP message that is used to ensure that no message is replayed in the network.

Stub Routers

You can use the EIGRP stub routing feature to improve network stability, reduce resource usage, and simplify stub router configuration. Stub routers connect to the EIGRP network through a remote router. See the Stub Routing section.

When using EIGRP stub routing, you need to configure the distribution and remote routers to use EIGRP and configure only the remote router as a stub. EIGRP stub routing does not automatically enable summarization on the distribution router. In most cases, you need to configure summarization on the distribution routers.

Without EIGRP stub routing, even after the routes that are sent from the distribution router to the remote router have been filtered or summarized, a problem might occur. For example, if a route is lost somewhere in the corporate network, EIGRP could send a query to the distribution router. The distribution router could then send a query to the remote router even if routes are summarized. If a problem communicating over the WAN link between the distribution router and the remote router occurs, EIGRP could get stuck in active condition and cause instability elsewhere in the network. EIGRP stub routing allows you to prevent queries to the remote router.

Route Summarization

You can configure a summary aggregate address for a specified interface. Route summarization simplifies route tables by replacing a number of more-specific addresses with an address that represents all the specific addresses. For example, you can replace 10.1.1.0/24, 10.1.2.0/24, and 10.1.3.0/24 with one summary address, 10.1.0.0/16.

If more specific routes are in the routing table, EIGRP advertises the summary address from the interface with a metric equal to the minimum metric of the more specific routes.

Note EIGRP does not support automatic route summarization.

Route Redistribution

You can use EIGRP to redistribute direct routes, static routes, routes learned by other EIGRP autonomous systems, or routes from other protocols. You configure route map with the redistribution to control which routes are passed into EIGRP. A route map allows you to filter routes based on attributes such as the destination, origination protocol, route type, route tag, and so on. See Configuring Route Policy Manager.

You also configure the default metric that is used for all imported routes into EIGRP.

Load Balancing

You can use load balancing to allow a router to distribute traffic over all the router network ports that are the same distance from the destination address. Load balancing increases the utilization of network segments, which increases effective network bandwidth.

Cisco NX-OS supports the Equal Cost Multiple Paths (ECMP) feature with up to 16 equal-cost paths in the EIGRP route table and the unicast RIB. You can configure EIGRP to load balance traffic across some or all of those paths.

Note EIGRP in Cisco NX-OS does not support unequal cost load balancing.

Split Horizon

You can use split horizon to ensure that EIGRP never advertises a route out of the interface where it was learned.

Split horizon is a method that controls the sending of EIGRP update and query packets. When you enable split horizon on an interface, Cisco NX-OS does not send update and query packets for destinations that were learned from this interface. Controlling update and query packets in this manner reduces the possibility of routing loops.

Split horizon with poison reverse configures EIGRP to advertise a learned route as unreachable back through that the interface that EIGRP learned the route from.

EIGRP uses split horizon or split horizon with poison reverse in the following scenarios:

- Exchanging topology tables for the first time between two routers in startup mode.
- Advertising a topology table change.
- Sending a query message.

By default, the split horizon feature is enabled on all interfaces.

Virtualization Support

Cisco NX-OS supports multiple instances of the EIGRP protocol that runs on the same system. EIGRP supports virtual routing and forwarding (VRF) instance. By default, Cisco NX-OS places you in the default VRF unless you specifically configure another VRF. See Configuring Layer 3 Virtualization.

By default, every instance uses the same system router ID. You can optionally configure a unique router ID for each instance.

Prerequisites for EIGRP

EIGRP has the following prerequisites:

- You must enable the EIGRP feature (see the Enabling the EIGRP Feature section).

Guidelines and Limitations

EIGRP has the following configuration guidelines and limitations:

- A metric configuration (either through the default-metric configuration option or through a route map) is required for redistribution from any other protocol, connected routes, or static routes (see Configuring Route Policy Manager).
- For graceful restart, an NSF-aware router must be up and completely converged with the network before it can assist an NSF-capable router in a graceful restart operation.
- For graceful restart, neighboring switches participating in the graceful restart must be NSF-aware or NSF-capable.
- Cisco NX-OS EIGRP is compatible with EIGRP in the Cisco IOS software.
- Do not change the metric weights without a good reason. If you change the metric weights, you must apply the change to all EIGRP routers in the same autonomous system.
- Consider using stubs for larger networks.
- Avoid redistribution between different EIGRP autonomous systems because the EIGRP vector metric will not be preserved.
- The **no ip next-hop-self** command does not guarantee reachability of the next-hop.
- The **ip passive-interface eigrp** command suppresses neighbors from forming.
- Cisco NX-OS does not support IGRP or connecting IGRP and EIGRP clouds.
- Autosummarization is not enabled by default.
- Cisco NX-OS supports only IP.

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

Default Settings

Following table lists the default settings for EIGRP parameters.

Table 13: Table 6-1 Default EIGRP Parameters

Parameters	Default
Administrative distance	• Internal routes- 90 • External routes—170
Bandwidth percent	50 percent
Default metric for redistributed routes	• bandwidth—100000 Kb/s • delay—100 (10 microsecond units) • reliability—255 • loading—1 • MTU—1500
EIGRP feature	Disabled
Hello interval	5 seconds
Hold time	15 seconds
Equal-cost paths	8
Metric weights	1 0 1 0 0
Next-hop address advertised	IP address of local interface
NSF convergence time	120
NSF route-hold time	240
NSF signal time	20
Redistribution	Disabled
Split horizon	Enabled

Configuring Basic EIGRP

This section includes the following topics:

Enabling the EIGRP Feature

You must enable the EIGRP feature before you can configure EIGRP.

Procedure

	Command or Action	Purpose

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

Step 2	**feature eigrp** **Example:** `switch(config)# feature eigrp`	Enables the EIGRP feature.
Step 3	(Optional) show feature **Example:** `switch(config)# show feature`	Displays information about enabled features.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

Use the **no feature eigrp** command to disable the EIGRP feature and remove all associated configuration.

Command	Purpose
no feature eigrp Example: `switch(config)# no feature eigrp`	Disables the EIGRP feature and removes all associated configuration.

Creating an EIGRP Instance

You can create an EIGRP instance and associate an interface with that instance. You assign a unique autonomous system number for this EIGRP process (see the Autonomous Systems section). Routes are not advertised or accepted from other autonomous systems unless you enable route redistribution.

Before you begin

Ensure that you have enabled the EIGRP feature (see the Enabling the EIGRP Feature section).

EIGRP must be able to obtain a router ID (for example, a configured loopback address) or you must configure the router ID option.

If you configure an instance tag that does not qualify as an AS number, you must configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.

Procedure

	Command or Action	Purpose

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

Step 2	**router eigrp** *instance-tag*
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Creates a new EIGRP process with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
		If you configure an *instance-tag* that does not qualify as an AS number, you must use the **autonomous-system** command to configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.

Step 3	(Optional) **autonomous-system** *as-number* **Example:** `switch(config-router)# autonomous-system 33`	Configures a unique AS number for this EIGRP instance. The range is from 1 to 65535.
Step 4	(Optional) **log-adjacency-changes** **Example:** `switch(config-router)# log-adjacency-changes`	Generates a system message whenever an adjacency changes state. This command is enabled by default.
Step 5	(Optional) **log-neighbor-warnings** [*seconds*] **Example:** `switch(config-router)# log-neighbor-warnings`	Generates a system message whenever a neighbor warning occurs. You can configure the time between warning messages, from 1 to 65535, in seconds. The default is 10 seconds. This command is enabled by default.

Step 6	**interface** *interface-type slot/port*
	Example:
	`switch(config-router)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode. Use **?** to determine the slot and port ranges.

Step 7	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 8	**ip router eigrp** *instance-tag* **Example:** `switch(config-if)# ip router eigrp Test1`	Associates this interface with the configured EIGRP process. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.

	Command or Action	Purpose
Step 9	**show ip eigrp interfaces** **Example:** `switch(config-if)# show ip eigrp interfaces`	Displays information about EIGRP interfaces.
Step 10	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

Use the **no router eigrp** command to remove the EIGRP process and the associated configuration.

Command	Purpose
no router eigrp *instance-tag* Example: `switch(config)# no router eigrp Test1`	Deletes the EIGRP process and all associated configuration.

Note You should also remove any EIGRP commands configured in interface mode if you remove the EIGRP process.

This example shows how to create an EIGRP process and configure an interface for EIGRP:

```
switch# configure terminal
switch(config)# router eigrp Test1
switch(config)# i nterface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# ip router eigrp Test1
switch(config-if)# no shutdown
switch(config-if)# copy running-config startup-config
```

For more information about other EIGRP parameters, see the Configuring Advanced EIGRP section.

Restarting an EIGRP Instance

You can restart an EIGRP instance. This clears all neighbors for the instance.

To restart an EIGRP instance and remove all associated neighbors, use the following commands:

Command	Purpose
flush-routes Example: `switch(config)# flush-routes`	Flushes all EIGRP routes in the unicast RIB when this EIGRP instance restarts.
restart eigrp *instance-tag* Example: `switch(config)# restart eigrp Test1`	Restarts the EIGRP instance and removes all neighbors. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.

Shutting Down an EIGRP Instance

You can gracefully shut down an EIGRP instance. This action moves all routes and adjacencies but preserves the EIGRP configuration.

To disable an EIGRP instance, use the following command in router configuration mode:

Command	Purpose
shutdown **Example:** `switch(config-router)# shutdown`	Disables this instance of EIGRP. The EIGRP router configuration remains.

Configuring Advanced EIGRP

This section includes the following topics:

Configuring Authentication in EIGRP

You can configure authentication between neighbors for EIGRP. See the Authentication section.

You can configure EIGRP authentication for the EIGRP process or for individual interfaces. Interface EIGRP authentication configuration overrides the EIGRP process-level authentication configuration.

Before you begin

Ensure that you have enabled the EIGRP feature (see the Enabling the EIGRP Feature section).

Ensure that all neighbors for an EIGRP process share the same authentication configuration, including the shared authentication key.

Create a keychain for this authentication configuration.

Procedure

	Command or Action	Purpose

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

Step 2	**router eigrp** *instance-tag*
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Creates a new EIGRP process with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
		If you configure an *instance-tag* that does not qualify as an AS number, you must use the **autonomous-system** command to configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.

Step 3	**address-family ipv4 unicast**
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters the address-family configuration mode. This command is optional for IPv4.

Step 4	**authentication keychain** *keychain* **Example:** `switch(config-router-af)# authentication keychain routeKeys`	Associates a keychain with this EIGRP process for this VRF. The keychain can be any case-sensitive, alphanumeric string up to 20 characters.
Step 5	**authentication mode md5** **Example:** `switch(config-router-af)# authentication mode md5`	Configures MD5 message digest authentication mode for this VRF.

Step 6	**interface** *interface-type slot/port*
	Example:
	`switch(config-router-af) interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode. Use **?** to find the supported interfaces.

Step 7	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 8	**ip router eigrp** *instance-tag* **Example:** `switch(config-if)# ip router eigrp Test1`	Associates this interface with the configured EIGRP process. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.

	Command or Action	Purpose
Step 9	**ip authentication keychain eigrp** instance-tag keychain **Example:** `switch(config-if)# ip authentication keychain eigrp Test1 routeKeys`	Associates a keychain with this EIGRP process for this interface. This configuration overrides the authentication configuration set in the router VRF mode. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
Step 10	**ip authentication mode eigrp** *instance-tag* **md5** **Example:** `switch(config-if)# ip authentication mode eigrp Test1 md5`	Configures the MD5 message digest authentication mode for this interface. This configuration overrides the authentication configuration set in the router VRF mode. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
Step 11	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure MD5 message digest authentication for EIGRP over Ethernet interface 1/2:

```
switch# configure terminal
switch(config)# router eigrp Test1
switch(config-router)# exit
switch(config)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# ip router eigrp Test1
switch(config-if)# ip authentication keychain eigrp Test1 routeKeys
switch(config-if)# ip authentication mode eigrp Test1 md5
switch(config-if)# copy running-config startup-config
```

Configuring EIGRP Stub Routing

To configure a router for EIGRP stub routing, use the following command in address-family configuration mode:

Command	Purpose
stub [**direct** | **receive-only** | **redistributed** [**direct**] **leak-map** *map-name*] Example: `switch(config-router-af)# eigrp stub redistributed`	Configures a remote router as an EIGRP stub router. The map name can be any case-sensitive, alphanumeric string up to 20 characters.

This example shows how to configure a stub router to advertise directly connected and redistributed routes:

```
switch# configure terminal

switch(config)# router eigrp Test1

switch(config-router)# address-family ipv4 unicast

switch(config-router-af)# stub direct redistributed

switch(config-router-af)# copy running-config startup-config
```

Use the show ip eigrp neighbor detail command to verify that a router has been configured as a stub router. The last line of the output shows the stub status of the remote or spoke router. This example shows the output from the show **ip eigrp neighbor detail** command:

```
Router# show ip eigrp neighbor detail
IP-EIGRP neighbors for process 201
H Address Interface Hold Uptime SRTT RTO Q Seq Type
(sec) (ms) Cnt Num
0 10.1.1.2 Se3/1 11 00:00:59 1 4500 0 7
Version 12.1/1.2, Retrans: 2, Retries: 0
Stub Peer Advertising ( CONNECTED SUMMARY) Routes
```

Configuring a Summary Address for EIGRP

You can configure a summary aggregate address for a specified interface. If any more specific routes are in the routing table, EIGRP will advertise the summary address out the interface with a metric equal to the minimum of all more specific routes. See the Route Summarization section.

To configure a summary aggregate address, use the following command in interface configuration mode:

Command	Purpose
ip summary-address eigrp *instance-tag ip-prefix/length* [*distance* | **leak-map** *map-name*] Example: `switch(config-if)# ip summary-address eigrp Test1 192.0.2.0/8`	Configures a summary aggregate address as either an IP address and network mask, or an IP prefix/length. The instance tag and map name can be any case-sensitive, alphanumeric string up to 20 characters. You can optionally configure the administrative distance for this aggregate address. The default administrative distance is 5 for aggregate addresses.

This example causes EIGRP to summarize network 192.0.2.0 out Ethernet 1/2 only:

```
switch(config)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# ip summary-address eigrp Test1 192.0.2.0 255.255.255.0
```

Redistributing Routes into EIGRP

You can redistribute routes in EIGRP from other routing protocols.

Before you begin

Ensure that you have enabled the EIGRP feature (see the Enabling the EIGRP Feature section).

You must configure the metric (either through the default-metric configuration option or through a route map) for routes redistributed from any other protocol.

You must create a route map to control the types of routes that are redistributed into EIGRP. See Configuring Route Policy Manager.

Procedure

	Command or Action	Purpose

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

Step 2	**router eigrp** *instance-tag*
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Creates a new EIGRP process with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
		If you configure an *instance-tag* that does not qualify as an AS number, you must use the **autonomous-system** command to configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.

Step 3	**address-family ipv4 unicast**
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters the address-family configuration mode. This command is optional for IPv4.

Step 4	**redistribute** { **bgp** *as* | {**eigrp** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **direct** | **static** } **route-map** *name* **Example:** `switch(config-router-af)# redistribute bgp 100 route-map BGPFilter`	Injects routes from one routing domain into EIGRP. The instance tag and map name can be any case-sensitive, alphanumeric string up to 20 characters.
Step 5	**default-metric** *bandwidth delay reliability loading mtu* **Example:** `switch(config-router-af)# default-metric 500000 30 200 1 1500`	Sets the metrics assigned to routes learned through route redistribution. The default values are as follows: • bandwidth—100000 Kb/s • delay—100 (10 microsecond units) • reliability—255 • loading—1 • MTU—1492
Step 6	**show ip eigrp route-map statistics redistribute** **Example:**	Displays information about EIGRP route map statistics.

	Command or Action	Purpose
	`switch(config-router-af)# show ip eigrp route-map statistics redistribute bgp`	
Step 7	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to redistribute BGP into EIGRP for IPv4:

```
switch# configure terminal
switch(config)# router eigrp Test1
switch(config-router)# redistribute bgp 100 route-map BGPFilter
switch(config-router)# default-metric 500000 30 200 1 1500
switch(config-router)# copy running-config startup-config
```

Limiting the Number of Redistributed Routes

Route redistribution can add many routes to the EIGRP route table. You can configure a maximum limit to the number of routes accepted from external protocols. EIGRP provides the following options to configure redistributed route limits:

- Fixed limit—Logs a message when EIGRP reaches the configured maximum. EIGRP does not accept any more redistributed routes. You can optionally configure a threshold percentage of the maximum where EIGRP will log a warning when that threshold is passed.
- Warning only—Logs a warning only when EIGRP reaches the maximum. EIGRP continues to accept redistributed routes.
- Withdraw—Start the timeout period when EIGRP reaches the maximum. After the timeout period, EIGRP requests all redistributed routes if the current number of redistributed routes is less than the maximum limit. If the current number of redistributed routes is at the maximum limit, EIGRP withdraws all redistributed routes. You must clear this condition before EIGRP accepts more redistributed routes.

  You can optionally configure the timeout period.

Before you begin

Ensure that you have enabled the EIGRP feature (see the Enabling the EIGRP Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router eigrp** *instance-tag*
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Enters global configuration mode.

Step 3	**redistribute** { **bgp** *id* | **direct** | **eigrp** *id* | **ospf** *id* | **rip** *id* | **static** } **route-map** *map-name* **Example:** `switch(config-router)# redistribute bgp route-map FilterExternalBGP`	Redistributes the selected protocol into EIGRP through the configured route map.
Step 4	**redistribute maximum-prefix** *max* [*threshold*] [**warning-only** | **withdraw** [*num-retries timeout*]] **Example:** `switch(config-router)# redistribute maximum-prefix 1000 75 warning-only`	Specifies a maximum number of prefixes that EIGRP will distribute. The range is from 0 to 65536. Optionally specifies the following: • **threshold** —Percent of maximum prefixes that will trigger a warning message. • **warning-only** —Logs an warning message when the maximum number of prefixes is exceeded. • **withdraw** —Withdraws all redistributed routes. Optionally tries to retrieve the redistributed routes. The *num-retries* range is from 1 to 12. The *timeout* is from 60 to 600 seconds. The default is 300 seconds. Use **clear ip eigrp redistribution** if all routes are withdrawn.
Step 5	(Optional) **show running-config eigrp** **Example:** `switch(config-router)# show running-config eigrp`	Displays the EIGRP configuration.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to limit the number of redistributed routes into EIGRP:

```
switch# configure terminal
switch(config)# router eigrp Test1
switch(config-router)# redistribute bgp route-map FilterExternalBGP
switch(config-router)# redistribute maximum-prefix 1000 75
```

Configuring Load Balancing in EIGRP

You can configure load balancing in EIGRP. You can configure the number of Equal Cost Multiple Path (ECMP) routes using the maximum paths option.

Before you begin

Ensure that you have enabled the EIGRP feature (see the Enabling the EIGRP Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router eigrp** *instance-tag*
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Creates a new EIGRP process with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
		If you configure an *instance-tag* that does not qualify as an AS number, you must use the **autonomous-system** command to configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.

Step 3	**address-family ipv4 unicast**
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters the address-family configuration mode. This command is optional for IPv4.

Step 4	**maximum-paths** *num-paths* **Example:** `switch(config-router-af)# maximum-paths 5`	Sets the number of equal cost paths that EIGRP will accept in the route table. The range is from 1 to 16. The default is 8.
Step 5	(Optional) **copy running-config startup-config** **Example:**	Saves this configuration change.

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

Example

This example shows how to configure equal cost load balancing for EIGRP over IPv4 with a maximum of six equal cost paths:

```
switch# configure terminal
switch(config)# router eigrp Test1
switch(config-router)# maximum-paths 6
switch(config-router)# copy running-config startup-config
```

Adjusting the Interval Between Hello Packets and the Hold Time

You can adjust the interval between hello messages and the hold time.

By default, hello messages are sent every 5 seconds. The hold time is advertised in hello messages and indicates to neighbors the length of time that they should consider the sender valid. The default hold time is three times the hello interval, or 15 seconds.

To change the interval between hello packets, use the following command in interface configuration mode:

Command	Purpose
ip hello-interval eigrp *instance-tag seconds* Example: `switch(config-if)# ip hello-interval eigrp Test1 30`	Configures the hello interval for an EIGRP routing process. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters. The range is from 1 to 65535 seconds. The default is 5.

On very congested and large networks, the default hold time might not be sufficient time for all routers to receive hello packets from their neighbors. In this case, you might want to increase the hold time.

To change the hold time, use the following command in interface configuration mode:

Command	Purpose
ip hold-time eigrp *instance-tag seconds* Example: `switch(config-if)# ip hold-time eigrp Test1 30`	Configures the hold time for an EIGRP routing process. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters. The range is from 1 to 65535.

Use the **show ip eigrp interface detail** command to verify timer configuration.

Disabling Split Horizon

You can use split horizon to block route information from being advertised by a router out of any interface from which that information originated. Split horizon usually optimizes communications among multiple routing switches, particularly when links are broken.

By default, split horizon is enabled on all interfaces.

To disable split horizon, use the following command in interface configuration mode:

Command	Purpose
no ip split-horizon eigrp *instance-tag* Example: `switch(config-if)# no ip split-horizon eigrp Test1`	Disables split horizon.

Tuning EIGRP

You can configure optional parameters to tune EIGRP for your network.

You can configure the following optional parameters in address-family configuration mode:

Command	Purpose
default-information originate [**always** | **route-map** *map-name*] Example: `switch(config-router-af)# default-information originate always`	Originates or accepts the default route with prefix 0.0.0.0/0. When a route map is supplied, the default route is originated only when the route map yields a true condition. The map name can be any case-sensitive, alphanumeric string up to 20 characters.
distance *internal external* Example: `switch(config-router-af)# distance 25 100`	Configures the administrative distance for this EIGRP process. The range is from 1 to 255. The internal value sets the distance for routes learned from within the same autonomous system (the default value is 90). The external value sets the distance for routes learned from an external autonomous system (the default value is 170).
metric maximum-hops *hop-count* Example: `switch(config-router-af)# metric maximum-hops 70`	Sets maximum allowed hops for an advertised route. Routes over this maximum are advertised as unreachable. The range is from 1 to 255. The default is 100.

Command	Purpose
metric weights *tos k1 k2 k3 k4 k5* Example: `switch(config-router-af)# metric weights 0 1 3 2 1 0`	Adjusts the EIGRP metric or K value. EIGRP uses the following formula to determine the total metric to the network: metric = [k1*bandwidth + (k2*bandwidth)/(256 – load) + k3*delay] * [k5/(reliability + k4)] Default values and ranges are as follows: • TOS—0. The range is from 0 to 8. • k1—1. The range is from 0 to 255. • k2—0. The range is from 0 to 255. • k3—1. The range is from 0 to 255. • k4—0. The range is from 0 to 255. • k5—0. The range is from 0 to 255.
timers active-time { *time-limit* | **disabled** } **Example** : `switch(config-router-af)# timers active-time 200.`	Sets the time the router waits in minutes (after sending a query) before declaring the route to be stuck in the active (SIA) state. The range is from 1 to 65535. The default is 3.

You can configure the following optional parameters in interface configuration mode:

Command	Purpose
ip band width eigrp *instance-tag bandwidth* Example: `switch(config-if)# ip bandwidth eigrp Test1 30000`	Configures the bandwidth metric for EIGRP on an interface. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters. The bandwidth range is from 1 to 2,560,000,000 Kb/s.
ip band width-percent eigrp *instance-tag percent* Example: `switch(config-if)# ip bandwidth-percent eigrp Test1 30`	Configures the percentage of bandwidth that EIGRP might use on an interface. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters. The percent range is from 0 to 100. The default is 50.
no ip delay eigrp *instance-tag delay* Example: `switch(config-if)# ip delay eigrp Test1 100`	Configures the delay metric for EIGRP on an interface. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters. The delay range is from 1 to 16777215 (in tens of microseconds).

Command	Purpose
ip distribute-list eigrp *instance-tag* { **prefix-list** *name* | **route-map** *name* } { **in** | **out** } Example: `switch(config-if)# ip distribute-list eigrp Test1 route-map EigrpTest in`	Configures the route filtering policy for EIGRP on this interface. The instance tag, prefix list name, and route map name can be any case-sensitive, alphanumeric string up to 20 characters.
no ip next-hop-self eigrp *instance-tag* Example: `switch(config-if)# ip next-hop-self eigrp Test1`	Configures EIGRP to use the received next-hop address rather than the address for this interface. The default is to use the IP address of this interface for the next-hop address. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
ip offset-list eigrp *instance-tag* { **prefix-list** *name* | **route-map** *name*} { **in** | **out** } *offset* Example: `switch(config-if)# ip offfset-list eigrp Test1 prefix-list EigrpList in`	Adds an offset to incoming and outgoing metrics to routes learned by EIGRP. The instance tag, prefix list name, and route map name can be any case-sensitive, alphanumeric string up to 20 characters.
ip passive-interface eigrp *instance-tag* **Example:** `switch(config-if)# ip passive-interface eigrp Test1`	Suppresses EIGRP hellos, which prevents neighbors from forming and sending routing updates on an EIGRP interface. The instance tag can be any case-sensitive, alphan

Configuring Virtualization for EIGRP

Ensure that you have enabled the EIGRP feature (see the Enabling the EIGRP Feature section).

Before you begin

You can create multiple VRFs and use the same or multiple EIGRP processes in each VRF. You assign an interface to a VRF.

Note Configure all other parameters for an interface after you configure the VRF for an interface. Configuring a VRF for an interface deletes all other configuration for that interface.

Procedure

	Command or Action	Purpose

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

	Command or Action	Purpose

Step 2	**vrf context** *vrf-name*
	Example:
	`switch(config)# vrf context RemoteOfficeVRF`
	`switch(config-vrf)#`
		Creates a new VRF and enters VRF configuration mode. The VRN name can be any case-sensitive, alphanumeric string up to 20 characters.

Step 3	**router eigrp** *instance-tag*
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Creates a new EIGRP process with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
		If you configure an *instance-tag* that does not qualify as an AS number, you must use the **autonomous-system** command to configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.

Step 4	**interface** *ethernet slot/port*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode. Use **?** to find the slot and port ranges.

Step 5	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 6	**vrf member** *vrf-name* **Example:** `switch(config-if)# vrf member RemoteOfficeVRF`	Adds this interface to a VRF. The VRF name can be any case-sensitive, alphanumeric string up to 20 characters.
Step 7	**ip router eigrp** *instance-tag* **Example:** `switch(config-if)# ip router eigrp Test1`	Adds this interface to the EIGRP process. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
Step 8	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to create a VRF and add an interface to the VRF:

```
switch# configure terminal
switch(config)# vrf context NewVRF
switch(config-vrf)# router eigrp Test1
switch(config-router)# i nterface ethernet 1/2
switch(config-if)# no switchport
```

```
switch(config-if)# ip router eigrp Test1
switch(config-if)# vrf member NewVRF
switch(config-if)# copy running-config startup-config
```

Verifying the EIGRP Configuration

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

Command	Purpose
show ip eigrp [*instance-tag*]	Displays a summary of the configured EIGRP processes.
show ip eigrp [*instance-tag*] **interfaces** [*type number*] [**brief**] [**detail**]	Displays information about all configured EIGRP interfaces.
show ip eigrp instance-tag neighbors [**type number**]	Displays information about all the EIGRP neighbors. Use this command to verify the EIGRP neighbor configuration.
show ip eigrp [*instance-tag*] **route** [ip-prefix/length] [active] [all-links] [detail-links] [pending] [summary] [zero-successors] [vrf vrf-name]	Displays information about all the EIGRP routes.
show **ip** eigrp [instance-tag] topology [ip-prefix/length] [active] [all-links] [detail-links] [pending] [summary] [zero-successors] [vrf vrf-name]	Displays information about the EIGRP topology table.
show running-configuration eigrp	Displays the current running EIGRP configuration.

Displaying EIGRP Statistics

To display EIGRP statistics, use the following commands:

Command	Purpose
show ip eigrp [*instance-tag*] **accounting** [**vrf** *vrf-name*]	Displays accounting statistics for EIGRP.
show ip eigrp [**instance-tag**] **route-map statistics redistribute**	Displays redistribution statistics for EIGRP.
show ip eigrp [*instance-tag*] **traffic** [**vrf** *vrf-name*]	Displays traffic statistics for EIGRP.

Configuration Examples for EIGRP

This example shows how to configure EIGRP:

```
feature eigrp
interface ethernet 1/2
no switchport
ip address 192.0.2.55/24
```

```
ip router eigrp Test1
no shutdown
router eigrp Test1
router-id 192.0.2.1
```

Related Topics

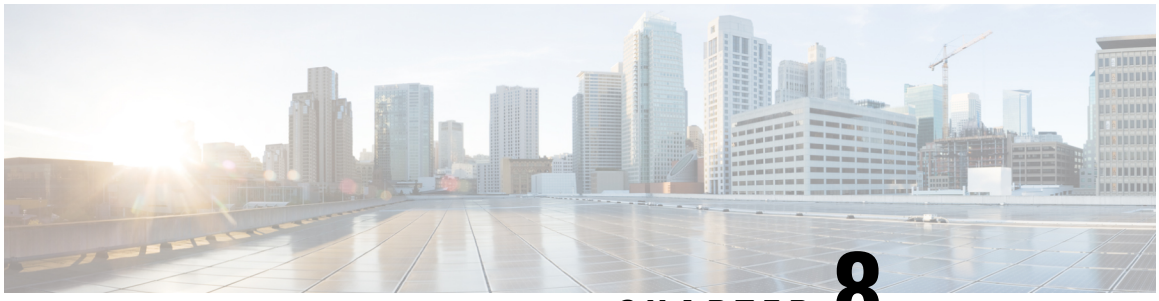
See Configuring Route Policy Manager for more information on route maps.

Additional References

For additional information related to implementing EIGRP, see the following sections:

MIBs

MIBs	MIBs Link
CISCO-EIGRP-MIB	To locate and download MIBs, go to the following: MIB Loc



CHAPTER 8

Configuring IS-IS

This chapter describes how to configure Integrated Intermediate System-to-Intermediate System (IS-IS) on the Cisco NX-OS device.

This chapter includes the following sections:

- About IS-IS, on page 163
- Prerequisites for IS-IS, on page 169
- Guidelines and Limitations for IS-IS, on page 169
- Default Settings, on page 169
- Configuring IS-IS, on page 170
- Verifying the IS-IS Configuration, on page 189
- Monitoring IS-IS, on page 190
- Configuration Examples for IS-IS, on page 191
- Related Topics, on page 191

About IS-IS

IS-IS is an Interior Gateway Protocol (IGP) based on Standardization (ISO)/International Engineering Consortium (IEC) 10589. Cisco NX-OS supports Internet Protocol version 4 (IPv4) and IPv6. IS-IS is a dynamic link-state routing protocol that can detect changes in the network topology and calculate loop-free routes to other nodes in the network. Each router maintains a link-state database that describes the state of the network and sends packets on every configured link to discover neighbors. IS-IS floods the link-state information across the network to each neighbor. The router also sends advertisements and updates on the link-state database through all the existing neighbors.

IS-IS Overview

IS-IS sends a hello packet out every configured interface to discover IS-IS neighbor routers. The hello packet contains information, such as the authentication, area, and supported protocols, which the receiving interface uses to determine compatibility with the originating interface. The hello packets are also padded to ensure that IS-IS establishes adjacencies only with interfaces that have matching maximum transmission unit (MTU) settings. Compatible interfaces form adjacencies, which update routing information in the link-state database through link-state update messages (LSPs). By default, the router sends a periodic LSP refresh every 10 minutes and the LSPs remain in the link-state database for 20 minutes (the LSP lifetime). If the router does not receive an LSP refresh before the end of the LSP lifetime, the router deletes the LSP from the database.

The LSP interval must be less than the LSP lifetime or the LSPs time out before they are refreshed.

IS-IS sends periodic hello packets to adjacent routers. If you configure transient mode for hello packets, these hello packets do not include the excess padding used before IS-IS establishes adjacencies. If the MTU value on adjacent routers changes, IS-IS can detect this change and send padded hello packets for a period of time. IS-IS uses this feature to detect mismatched MTU values on adjacent routers. For more information, see the Configuring the Transient Mode for Hello Padding section.

IS-IS Areas

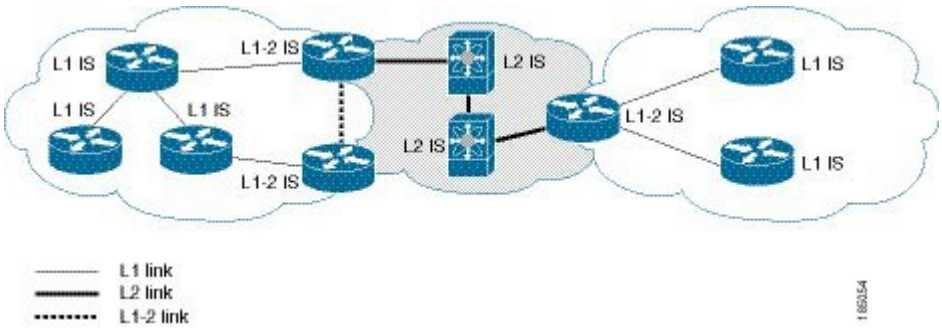
You can design IS-IS networks as a single area that includes all routers in the network or as multiple areas that connect into a backbone or Level 2 area. Routers in a nonbackbone area are Level 1 routers that establish adjacencies within a local area (intra-area routing). Level 2 area routers establish adjacencies to other Level 2 routers and perform routing between Level 1 areas (inter-area routing). A router can have both Level 1 and Level 2 areas configured. These Level 1/Level 2 routers act as area border routers that route information from the local area to the Level 2 backbone area (see Figure 7-1).

Within a Level 1 area, routers know how to reach all other routers in that area. The Level 2 routers know how to reach other area border routers and other Level 2 routers. Level 1/Level 2 routers straddle the boundary between two areas, routing traffic to and from the Level 2 backbone area. Level1/Level2 routers use the attached (ATT) bit signal Level 1 routers to set a default route to this Level1/Level2 router to connect to the Level 2 area.

In some instances, such as when you have two or more Level1/Level 2 routers in an area, you may want to control which Level1/Level2 router that the Level 1 routers use as the default route to the Level 2 area. You can configure which Level1/Level2 router sets the attached bit. For more information, see the Verifying the IS-IS Configuration section.

Each IS-IS instance in Cisco NX-OS supports either a single Level 1 or Level 2 area, or one of each. By default, all IS-IS instances automatically support Level 1 and Level 2 routing.

Figure 25: IS-IS Network Divided into Areas



An autonomous system boundary router (ASBR) advertises external destinations throughout the IS-IS autonomous system. External routes are the routes redistributed into IS-IS from any other protocol.

NET and System ID

Each IS-IS instance has an associated network entity title (NET). The NET is comprised of the IS-IS system ID, which uniquely identifies this IS-IS instance in the area and the area ID. For example, if the NET is 47.0004.004d.0001.0001.0c11.1111.00, the system ID is 0001.0c11.1111.00 and the area is ID 47.0004.004d.0001.

Designated Intermediate System

IS-IS uses a designated intermediate system (DIS) in broadcast networks to prevent each router from forming unnecessary links with every other router on the broadcast network. IS-IS routers send LSPs to the DIS, which manages all the link-state information for the broadcast network. You can configure the IS-IS priority that IS-IS uses to select the DIS in an area.

Note No DIS is required on a point-to-point network.

IS-IS Authentication

You can configure authentication to control adjacencies and the exchange of LSPs. Routers that want to become neighbors must exchange the same password for their configured level of authentication. IS-IS blocks a router that does not have the correct password. You can configure IS-IS authentication globally or for an individual interface for Level 1, Level 2, or both Level 1/Level 2 routing.

IS-IS supports the following authentication methods:

- Clear text—All packets exchanged carry a cleartext 128-bit password.
- MD5 digest—All packets exchanged carry a message digest that is based on a 128-bit key.

To provide protection against passive attacks, IS-IS never sends the MD5 secret key as cleartext through the network. In addition, IS-IS includes a sequence number in each packet to protect against replay attacks.

You can also use keychains for hello and LSP authentication. See the *Cisco Nexus 3600 NX-OS Security Configuration Guide* for information on keychain management.

Mesh Groups

A mesh group is a set of interfaces in which all routers reachable over the interfaces have at least one link to every other router. Many links can fail without isolating one or more routers from the network.

In normal flooding, an interface receives a new LSP and floods the LSP out over all other interfaces on the router. With mesh groups, when an interface that is part of a mesh group receives a new LSP, the interface does not flood the new LSP over the other interfaces that are part of that mesh group.

Note You may want to limit LSPs in certain mesh network topologies to improve network scalability. Limiting LSP floods might also reduce the reliability of the network (in case of failures). For this reason, we recommend that you use mesh groups only if specifically required, and then only after you make a careful network design.

You can also configure mesh groups in block mode for parallel links between routers. In this mode, all LSPs are blocked on that interface in a mesh group after the routers initially exchange their link-state information.

Overload Bit

IS-IS uses the overload bit to tell other routers not to use the local router to forward traffic but to continue routing traffic destined for that local router.

You may want to use the overload bit in these situations:

- The router is in a critical condition.
- Graceful introduction and removal of the router to/from the network.
- Other (administrative or traffic engineering) reasons such as waiting for BGP convergence.

Route Summarization

You can configure a summary aggregate address. Route summarization simplifies route tables by replacing a number of more-specific addresses with an address that represents all the specific addresses. For example, you can replace 10.1.1.0/24, 10.1.2.0/24, and 10.1.3.0/24 with one summary address, 10.1.0.0/16.

If more specific routes are in the routing table, IS-IS advertises the summary address with a metric equal to the minimum metric of the more specific routes.

Note Cisco NX-OS does not support automatic route summarization.

Configuring Redistribution

You can configure IS-IS to accept routing information from another routing protocol and redistribute that information through the IS-IS network. You can optionally assign a default route for redistributed routes.

Before you begin

You must enable IS-IS (see the Enabling the IS-IS Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router isis** *instance-tag*
	Example:
	`switch(config)# router isis Enterprise`
	`switch(config-router)#`
		Creates a new IS-IS instance with the configured *instance tag*.

Step 3	**address-family** {**ipv4** | **ipv6**} **unicast**
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters address family configuration mode.

	Command or Action	Purpose
Step 4	**redistribute** { **bgp** *as* | **direct** |{ **eigrp** | **isis** | **ospf** | **ospfv3** | **rip** } *instance-tag* | **static** } **route-map** *map-name* **Example:** `switch(config-router-af)# redistribute eigrp 201 route-map ISISmap`	Redistributes routes from other protocols into IS-IS. See the Configuring Route Maps section for more information about route maps.
Step 5	(Optional) **default-information originate** [**always**] [**route-map** *map-name*] **Example:** `switch(config-router-af)# default-information originate always`	Generates a default route into IS-IS.
Step 6	(Optional) **distribute** { level-1 | level-2 } into { level-1 | level-2 } { **route-map** *route-map* | **all** } **Example:** `switch(config-router-af)# distribute level-1 into level-2 all`	Redistributes routes from one IS-IS level to the other IS-IS level.
Step 7	(Optional) **show isis** [**vrf** *vrf-name*] { **ip** | **ipv6**} **route** *ip-prefix* [*detail* | **longer-prefixes** [**summary** | **detail**]] **Example:** `switch(config-if)# show isis ip summary-address`	Shows the IS-IS routes.
Step 8	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to redistribute EIGRP into IS-IS:

```
switch# configure terminal
switch(config)# router isis Enterprise
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# redistribute eigrp 201 route-map ISISmap
switch(config-router-af)# copy running-config startup-config
```

Load Balancing

You can use load balancing to allow a router to distribute traffic over all the router network ports that are the same distance from the destination address. Load balancing increases the utilization of network segments and increases the effective network bandwidth.

Cisco NX-OS supports the Equal Cost Multiple Paths (ECMP) feature with up to 64 equal-cost paths in the IS-IS route table and the unicast RIB. You can configure IS-IS to load balance traffic across some or all of those paths.

BFD

This feature supports bidirectional forwarding detection (BFD) for IPv4 and IPv6. BFD is a detection protocol designed to provide fast forwarding-path failure detection times. BFD provides subsecond failure detection between two adjacent devices and can be less CPU-intensive than protocol hello messages because some of the BFD load can be distributed onto the data plane on supported modules. See the Cisco Nexus 3600 NX-OS Interfaces Configuration Guide for more information.

Virtualization Support

Cisco NX-OS supports multiple process instances for IS-IS. Each IS-IS instance can support multiple virtual routing and forwarding (VRF) instances, up to the system limit. For the number of supported IS-IS instances, see the Cisco Nexus 3600 NX-OS Verified Scalability Guide.

High Availability and Graceful Restart

Cisco NX-OS provides a multilevel high-availability architecture. IS-IS supports stateful restart, which is also referred to as non-stop routing (NSR). If IS-IS experiences problems, it attempts to restart from its previous run-time state. The neighbors would not register any neighbor event in this case. If the first restart is not successful and another problem occurs, IS-IS attempts a graceful restart as per RFC 3847. A graceful restart, or non-stop forwarding (NSF), allows IS-IS to remain in the data forwarding path through a process restart. When the restarting IS-IS interface is operational again, it rediscovers its neighbors, establishes adjacency, and starts sending its updates again. At this point, the NSF helpers recognize that the graceful restart has finished.

A stateful restart is used in the following scenarios:

- First recovery attempt after process experiences problems
- User-initiated switchover using the **system switchover** command

A graceful restart is used in the following scenarios:

- Second recovery attempt after the process experiences problems within a 4-minute interval
- Manual restart of the process using the **restart isis** command
- Active supervisor removal
- Active supervisor reload using the **reload module** *active-sup* command

Note Graceful restart is on by default, and we strongly recommend that you do not disable it.

Multiple IS-IS Instances

Cisco NX-OS supports multiple instances of the IS-IS protocol that run on the same node. You cannot configure multiple instances over the same interface. Every instance uses the same system router ID. For the number of supported IS-IS instances, see the Cisco Nexus 3600 NX-OS Verified Scalability Guide.

Prerequisites for IS-IS

IS-IS has the following prerequisites:

- You must enable IS-IS (see the Enabling the IS-IS Feature section).

Guidelines and Limitations for IS-IS

IS-IS has the following configuration guidelines and limitations:

- Because the default reference bandwidth is different for Cisco NX-OS and Cisco IOS, the advertised tunnel IS-IS metric is different for these two operating systems.

Default Settings

The following table lists the default settings for IS-IS parameters.

Table 14: Default IS-IS Parameters

Parameters	Default
Administrative distance	115
Area level	Level-1-2
DIS priority	64
Graceful restart	Enabled
Hello multiplier	3
Hello padding	Enabled
Hello time	10 seconds
IS-IS feature	Disabled
LSP interval	33
LSP MTU	1492

Parameters	Default
Maximum LSP lifetime	1200 seconds
Maximum paths	8
Metric	40
Reference bandwidth	40 Gbps

Configuring IS-IS

IS-IS Configuration Modes

The following sections show how to enter each of the configuration modes. From a mode, you can enter the **?** command to display the commands available in that mode.

Router Configuration Mode

This example shows how to enter router configuration mode:

```
switch#: configure terminal
switch(config)# router isis isp
switch(config-router)#
```

Router Address Family Configuration Mode

This example shows how to enter router address family configuration mode:

```
switch(config)# router isis isp
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)#
```

Enabling the IS-IS Feature

You must enable the IS-IS feature before you can configure IS-IS.

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 the IS-IS feature.

	Command or Action	Purpose
Step 3	(Optional) **show feature** **Example:** `switch(config)# show feature`	Displays enabled and disabled features.
Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

To disable the IS-IS feature and remove all associated configuration, use the following command in configuration mode:

Command	Purpose
no feature isis Example: `switch(config)# no feature isis`	Disables the IS-IS feature and removes all associated configurations.

Creating an IS-IS Instance

You can create an IS-IS instance and configure the area level for that instance.

Before you begin

You must enable IS-IS (see the Enabling the IS-IS Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router isis** *instance-tag*
	Example:
	`switch(config)# router isis Enterprise`
	`switch(config-router)#`
		Creates a new IS-IS instance with the configured *instance tag*.

Step 3	**net** *network-entity-title* **Example:**	Configures the NET for this IS-IS instance.

	Command or Action	Purpose
	`switch(config-router)# net 47.0004.004d.0001.0001.0c11.1111.00`	
Step 4	(Optional) **is-type** { **level-1** | **level-2** | **level-1-2** } **Example:** `switch(config-router)# is-type level-2`	Configures the area level for this IS-IS instance. The default is level-1-2.
Step 5	(Optional) **show isis** [**vrf** *vrf-name*] **process** **Example:** `switch(config)# show isis process`	Displays a summary of IS-IS information for all IS-IS instances.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

To remove the IS-IS instance and the associated configuration, use the following command in configuration mode:

Command	Purpose
no router isis *instance-tag*	Deletes the IS-IS instance and all associated configurations.

Note You must also remove any IS-IS commands that are configured in interface mode to completely remove all configurations for the IS-IS instance.

You can configure the following optional parameters for IS-IS:

Command	Purpose
distance *value* Example: `switch(config-router)# distance 30`	Sets the administrative distance for IS-IS. The range is from 1 to 255. The default is 115.
log-adjacency-changes Example: `switch(config-router)# log-adjacency-changes`	Sends a system message whenever an IS-IS neighbor changes the state.

Command	Purpose
lsp-mtu *size* Example: `switch(config-router)# lsp-mtu 600`	Sets the MTU for LSPs in this IS-IS instance. The range is from 128 to 4352 bytes. The default is 1492.
maximum-paths *number* **Example:** `switch(config-router)# maximum-paths 6`	Configures the maximum number of equal-cost paths that IS-IS maintains in the route table. The range is from 1 to 64. The default is 8.
reference-bandwidth *bandwidth-value* { **Mbps** | **Gbps** } Example: `switch(config-router)# reference-bandwidth 100 Gbps`	Sets the default reference bandwidth used for calculating the IS-IS cost metric. The range is from 1 to 4000 Gbps. The default is 40 Gbps.

This example shows how to create an IS-IS instance in a level 2 area:

```
switch# configure terminal
switch(config)# router isis Enterprise
switch(config-router)# net 47.0004.004d.0001.0001.0c11.1111.00
switch(config-router)# is-type level 2
switch(config-router)# copy running-config startup-config
```

To clear neighbor statistics and remove adjacencies, use the following command in router configuration mode:

Command	Purpose
clear isis [*instance-tag*] **adjacency** [***** | *system-id* | *interface*] Example: `switch(config-if)# clear isis adjacency *`	Clears neighbor statistics and removed adjacencies for this IS-IS instance.

Restarting an IS-IS Instance

You can restart an IS-IS instance. This action clears all neighbors for the instance.

To restart an IS-IS instance and remove all associated neighbors, use the following command:

Procedure

	Command or Action	Purpose
Step 1	restart isis instance-tag **Example:** `switch(config)# restart isis Enterprise`	Restarts the IS-IS instance and removes all neighbors.

Shutting Down IS-IS

You can shut down the IS-IS instance. This action disables this IS-IS instance and retains the configuration.

To shut down the IS-IS instance, use the following command in router configuration mode:

Procedure

	Command or Action	Purpose
Step 1	**shutdown** **Example:** `switch(config-router)# shutdown`	Disables the IS-IS instance.

Configuring IS-IS Authentication in an Area

You can configure IS-IS to authenticate LSPs in an area.

Before you begin

You must enable IS-IS. See Enabling the IS-IS Feature.

You must configure the keychain in global configuration mode if you reference it from the IS-IS configuration. See "Configuring Keychain Management" in the Cisco Nexus 9000 Series NX-OS Security Configuration Guide.

Procedure

	Command or Action	Purpose

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

Step 2	**router isis** *instance-tag*
	Example:
	`switch(config)# router isis Enterprise`
	`switch(config-router)#`
		Creates a new IS-IS instance with the configured *instance tag*.

Step 3	**authentication-type** { **cleartext** | **md5** } { **level-1** | **level-2** }
	Example:
	`switch(config-router)#`
	`authentication-type cleartext level-2`
		Sets the authentication method used for a Level 1 or Level 2 area as cleartext or as an MD5 authentication digest.

Step 4	**authentication key-chain** *key* { **level-1** | **level-2** } **Example:**	Configures the authentication key that is used for an IS-IS area-level authentication.

	Command or Action	Purpose
	`authentication key-chain key { level-1 | level-2 }`	
Step 5	(Optional) **authentication-check** { **level-1** | **level-2** } **Example:** `switch(config-router)# authentication-check level-2`	Enables checking the authentication parameters in a received packet.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure cleartext authentication on an IS-IS instance:

```
switch# configure terminal
switch(config)# router isis Enterprise
switch(config-router)# authentication-type cleartext level-2
switch(config-router)# authentication key-chain ISISKey level-2
switch(config-router)# copy running-config startup-config
```

Configuring IS-IS Authentication on an Interface

You can configure IS-IS to authenticate Hello packets on an interface.

Before you begin

You must enable IS-IS (see the Enabling the IS-IS Feature section).

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal switch(config)#`	Enters global configuration mode.
Step 2	**interface** *interface-type slot/port* **Example:** `switch(config)# interface ethernet 1/2 switch(config-if)#`	Enters interface configuration mode.
Step 3	**isis authentication-type** { **cleartext** | **md5** } { **level-1** | **level-2** } **Example:**	Sets the authentication type for IS-IS on this interface as cleartext or as an MD5 authentication digest.

	Command or Action	Purpose
	`switch(config-if)# isis authentication-type cleartext level-2`	
Step 4	**isis authentication key-chain** *key* { **level-1** | **level-2** } **Example:** `switch(config-if)# isis authentication key-chain ISISKey level-2`	Configures the authentication key used for IS-IS on this interface.
Step 5	(Optional) **isis authentication-check** { **level-1** | **level-2**} **Example:** `switch(config-if)# isis authentication-check`	Enables checking the authentication parameters in a received packet.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure cleartext authentication on an IS-IS instance:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# isis authentication-type cleartext level-2
switch(config-if)# isis authentication key-chain ISISKey
switch(config-if)# copy running-config startup-config
```

Configuring a Mesh Group

You can add an interface to a mesh group to limit the amount of LSP flooding for interfaces in that mesh group. You can optionally block all LSP flooding on an interface in a mesh group.

To add an interface to a mesh group, use the following command in interface configuration mode:

Procedure

	Command or Action	Purpose
Step 1	**isis mesh-group** { **blocked** | mesh-id } **Example:** `switch(config-if)# isis mesh-group 1`	Adds this interface to a mesh group. The range is from 1 to 4294967295.

Configuring a Designated Intermediate System

You can configure a router to become the designated intermediate system (DIS) for a multiaccess network by setting the interface priority.

To configure the DIS, use the following command in interface configuration mode:

Procedure

	Command or Action	Purpose
Step 1	**isis priority** *number* { **level-1** | **level-2** } **Example:** `switch(config-if)# isis priority 100 level-1`	Sets the priority for DIS selection. The range is from 0 to 127. The default is 64.

Configuring Dynamic Host Exchange

You can configure IS-IS to map between the system ID and the hostname for a router using dynamic host exchange.

To configure dynamic host exchange, use the following command in router configuration mode:

Procedure

	Command or Action	Purpose
Step 1	**hostname dynamic** **Example:** `switch(config-router)# hostname dynamic`	Enables dynamic host exchange.

Setting the Overload Bit

You can configure the router to signal other routers not to use this router as an intermediate hop in their shortest path first (SPF) calculations. You can optionally configure the overload bit temporarily on startup, until BGP converges.

In addition to setting the overload bit, you might also want to suppress certain types of IP prefix advertisements from LSPs for Level 1 or Level 2 traffic.

To set the overload bit, use the following command in router configuration mode:

Procedure

	Command or Action	Purpose
Step 1	**set-overload-bit** { **always** | **on-startup** { *seconds* | **wait-for bgp** *as-number* }} [**suppress** [**interlevel** | **external**]] **Example:**	Sets the overload bit for IS-IS. The seconds range is from 5 to 86400.

	Command or Action	Purpose
	`switch(config-router)# set-overload-bit on-startup 30`	

Configuring the Attached Bit

You can configure the attached bit to control which Level 1/Level 2 router that the Level 1 routers use as the default route to the Level 2 area. If you disable setting the attached bit, the Level 1 routers do not use this Level 1/Level 2 router to reach the Level 2 area.

To configure the attached bit for a Level 1/Level 2 router, use the following command in router configuration mode:

Procedure

	Command or Action	Purpose
Step 1	[**no**] **attached-bit** **Example:** `switch(config-router)# no attached-bit`	Configures the Level 1/Level 2 router to set the attached bit. This feature is enabled by default.

Configuring the Transient Mode for Hello Padding

You can configure the transient mode for hello padding to pad hello packets when IS-IS establishes adjacency and remove that padding after IS-IS establishes adjacency.

To configure the mode for hello padding, use the following command in router configuration mode:

Command	Purpose
[**no**] **isis hello-padding** **Example** : `switch(config-if)# no isis hello-padding`	Pads the hello packet to the full maximum transmission unit (MTU). The default is enabled. Use the **no** form of this command to configure the transient mode of hello padding.

Configuring a Summary Address

You can create aggregate addresses that are represented in the routing table by a summary address. One summary address can include multiple groups of addresses for a given level. Cisco NX-OS advertises the smallest metric of all the more-specific routes.

Before you begin

You must enable IS-IS (see the Enabling the IS-IS Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router isis** *instance-tag*
	Example:
	`switch(config)# router isis Enterprise`
	`switch(config-router)#`
		Creates a new IS-IS instance with the configured *instance tag*.

Step 3	**address-family** {**ipv4** | **ipv6**} **unicast**
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters address family configuration mode.

Step 4	**summary-address** *ip-prefix/mask-len* { **level-1** | **level-2** | **level-1-2** } **Example:** `switch(config-router-af)# summary-address 192.0.2.0/24 level-2`	Configures a summary address for an IS-IS area for IPv4 or IPv6 addresses.
Step 5	(Optional) **show isis** [**vrf** *vrf-name*] { **ip** | **ipv6**} **summary-address** *ip-prefix* [**longer-prefixes**] **Example:** `switch(config-if)# show isis ip summary-address`	Displays IS-IS IPv4 or IPv6 summary address information.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure an IPv4 unicast summary address for IS-IS:

```
switch# configure terminal
switch(config)# router isis Enterprise
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# summary-address 192.0.2.0/24 level-2
switch(config-router-af)# copy running-config startup-config
```

Configuring Redistribution

You can configure IS-IS to accept routing information from another routing protocol and redistribute that information through the IS-IS network. You can optionally assign a default route for redistributed routes.

Before you begin

You must enable IS-IS (see the Enabling the IS-IS Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router isis** *instance-tag*
	Example:
	`switch(config)# router isis Enterprise`
	`switch(config-router)#`
		Creates a new IS-IS instance with the configured *instance tag*.

Step 3	**address-family** {**ipv4** | **ipv6**} **unicast**
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters address family configuration mode.

Step 4	**redistribute** { **bgp** *as* | **direct** |{ **eigrp** | **isis** | **ospf** | **ospfv3** | **rip** } *instance-tag* | **static** } **route-map** *map-name* **Example:** `switch(config-router-af)# redistribute eigrp 201 route-map ISISmap`	Redistributes routes from other protocols into IS-IS. See the Configuring Route Maps section for more information about route maps.
Step 5	(Optional) **default-information originate** [**always**] [**route-map** *map-name*] **Example:** `switch(config-router-af)# default-information originate always`	Generates a default route into IS-IS.
Step 6	(Optional) **distribute** { level-1 | level-2 } into { level-1 | level-2 } { **route-map** *route-map* | **all** } **Example:** `switch(config-router-af)# distribute level-1 into level-2 all`	Redistributes routes from one IS-IS level to the other IS-IS level.

	Command or Action	Purpose
Step 7	(Optional) **show isis** [**vrf** *vrf-name*] { **ip** | **ipv6**} **route** *ip-prefix* [*detail* | **longer-prefixes** [**summary** | **detail**]] **Example:** `switch(config-if)# show isis ip summary-address`	Shows the IS-IS routes.
Step 8	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to redistribute EIGRP into IS-IS:

```
switch# configure terminal
switch(config)# router isis Enterprise
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# redistribute eigrp 201 route-map ISISmap
switch(config-router-af)# copy running-config startup-config
```

Limiting the Number of Redistributed Routes

Route redistribution can add many routes to the IS-IS route table. You can configure a maximum limit to the number of routes accepted from external protocols. IS-IS provides the following options to configure redistributed route limits:

- Fixed limit—Logs a message when IS-IS reaches the configured maximum. IS-IS does not accept any more redistributed routes. You can optionally configure a threshold percentage of the maximum where IS-IS logs a warning when that threshold is passed.
- Warning only—Logs a warning only when IS-IS reaches the maximum. IS-IS continues to accept redistributed routes.
- Withdraw—Starts the timeout period when IS-IS reaches the maximum. After the timeout period, IS-IS requests all redistributed routes if the current number of redistributed routes is less than the maximum limit. If the current number of redistributed routes is at the maximum limit, IS-IS withdraws all redistributed routes. You must clear this condition before IS-IS accepts more redistributed routes. You can optionally configure the timeout period.

Before you begin

You must enable IS-IS (see the Enabling the IS-IS Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router isis** *instance-tag*
	Example:
	`switch(config)# router isis Enterprise`
	`switch(config-router)#`
		Creates a new IS-IS instance with the configured instance tag.

Step 3	**redistribute** { **bgp** *id* | **direct** | **eigrp** *id* | **isis** *id* | **ospf** *id* | **rip** *id* | **static** } **route-map** *map-name* **Example:** `switch(config-router)# redistribute bgp route-map FilterExternalBGP`	Redistributes the selected protocol into IS-IS through the configured route map.
Step 4	**redistribute maximum-prefix** *max* [*threshold*] [**warning-only** | **withdraw** [*num-retries timeout*]] **Example:** `switch(config-router)# redistribute maximum-prefix 1000 75 warning-only`	Specifies a maximum number of prefixes that IS-IS distributes. The range is from 1 to 65535. You can optionally specify the following: • *threshold* —Percentage of maximum prefixes that triggers a warning message. • **warning-only** —Logs a warning message when the maximum number of prefixes is exceeded. • **withdraw** —Withdraws all redistributed routes. You can optionally try to retrieve the redistributed routes. The *num-retries* range is from 1 to 12. The *timeout* is 60 to 600 seconds. The default is 300 seconds. Use the **clear isis redistribution** command if all routes are withdrawn.
Step 5	(Optional) **show running-config isis** **Example:** `switch(config-router)# show running-config isis`	Displays the IS-IS configuration.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to limit the number of redistributed routes into IS-IS:

```
switch# configure terminal
switch(config)# router eigrp isis Enterprise
switch(config-router)# redistribute bgp route-map FilterExternalBGP
switch(config-router)# redistribute maximum-prefix 1000 75
```

Disabling Strict Adjacency Mode

When both IPv4 and IPv6 address families are enabled, strict adjacency mode is enabled by default. In this mode, the device does not form an adjacency with any router that does not have both address families enabled. You can disable strict adjacency mode using the no adjacency-check command.

Before you begin

You must enable IS-IS (see the Enabling the IS-IS Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router isis** *instance-tag*
	Example:
	`switch(config)# router isis Enterprise`
	`switch(config-router)#`
		Creates a new IS-IS instance with the configured instance tag.

Step 3	**address-family ipv4 unicast**
	Example:
	`switch(config-router)# address-family ipv4`
	`switch(config-router-af)#`
		Enters address family configuration mode.

Step 4	**no adjacency-check** **Example:** `switch(config-router-af)# no adjacency-check`	Disables strict adjacency mode for the IPv4 address family.

Step 5	**exit**
	Example:
	`switch(config-router-af)# exit`
	`switch(config-router)#`
		Exits address family configuration mode.

	Command or Action	Purpose

Step 6	**address-family ipv6 unicast**
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		Enters address family configuration mode.

Step 7	(Optional) **no adjacency-check** **Example:** `switch(config-router-af)# no adjacency-check`	Disables strict adjacency mode for the IPv6 address family.
Step 8	**show running-config isis** **Example:** `switch(config-router-af)# show running-config isis`	Displays the IS-IS configuration.
Step 9	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-af)# copy running-config startup-config`	Saves this configuration change.

Configuring a Graceful Restart

You can configure a graceful restart for IS-IS.

Before you begin

You must enable IS-IS (see the Enabling the IS-IS Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router isis** *instance-tag*
	Example:
	`switch(config)# router isis Enterprise`
	`switch(config-router)#`
		Creates a new IS-IS process with the configured name.

Step 3	**graceful-restart** **Example:** `switch(config-router)# graceful-restart`	Configures the graceful restart T3 timer. The range is from 30 to 65535 seconds. The default is 60.

	Command or Action	Purpose
Step 4	**graceful-restart t3 manual** *time* **Example:** `switch(config-router)# graceful-restart t3 manual 300`	Configures the graceful restart T3 timer. The range is from 30 to 65535 seconds. The default is 60.
Step 5	(Optional) **show running-config isis** **Example:** `switch(config-router)# show running-config isis`	Displays the IS-IS configuration.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to enable a graceful restart:

```
switch# configure terminal
switch(config)# router isis Enterprise
switch(config-router)# graceful-restart
switch(config-router)# copy running-config startup-config
```

Configuring Virtualization

You can configure multiple IS-IS instances and multiple VRFs and use the same or multiple IS-IS instances in each VRF. You assign an IS-IS interface to a VRF.

You must configure a NET for the configured VRF.

Note Configure all other parameters for an interface after you configure the VRF for an interface. Configuring a VRF for an interface deletes all the configuration for that interface.

Before you begin

You must enable IS-IS (see the Enabling the IS-IS Feature section).

Procedure

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

	Command or Action	Purpose

Step 2	**vrf context** *vrf-name*
	Example:
	`switch(config)# vrf context RemoteOfficeVRF`
	`switch(config-vrf)#`
		Creates a new VRF and enters VRF configuration mode.

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

Step 4	**router isis** *instance-tag*
	Example:
	`switch(config)# router isis Enterprise`
	`switch(config-router)#`
		Creates a new IS-IS instance with the configured instance tag.

Step 5	**vrf** *vrf-name*
	Example:
	`switch(config-router)# vrf RemoteOfficeVRF`
	`switch(config-router-vrf)#`
		Enters VRF configuration mode.

Step 6	**net** *network-entity-title* **Example:** `switch(config-router-vrf)# net 47.0004.004d.0001.0001.0c11.1111.00`	Configures the NET for this IS-IS instance.

Step 7	**exit**
	Example:
	`switch(config-router-vrf)# exit`
	`switch(config-router)#`
		Exits router VRF configuration mode.

Step 8	**interface** *ethernet slot/port*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 9	**vrf member** *vrf-name* **Example:** `switch(config-if)# vrf member RemoteOfficeVRF`	Adds this interface to a VRF.
Step 10	{ **ip** | **ipv6** } **address** *ip-prefix/length* **Example:** `switch(config-if)# ip address 192.0.2.1/16`	Configures an IP address for this interface. You must do this step after you assign this interface to a VRF.

	Command or Action	Purpose
Step 11	{ **ip** | **ipv6**} **router isis** *instance-tag* **Example:** `switch(config-if)# ip router isis Enterprise`	Associates this IPv4 or IPv6 interface with an IS-IS instance.
Step 12	(Optional) **show isis** [**vrf** *vrf-name*] [*instance-tag*] **interface** [*interface-type slot/port*] **Example:** `switch(config-if)# show isis Enterprise ethernet 1/2`	Displays IS-IS information for an interface in a VRF.
Step 13	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to create a VRF and add an interface to the VRF:

```
switch# configure terminal
switch(config)# vrf context NewVRF
switch(config-vrf)# exit
switch(config)# router isis Enterprise
switch(config-router)# vrf NewVRF
switch(config-router-vrf)# net 47.0004.004d.0001.0001.0c11.1111.00
switch(config-router-vrf)# interface ethernet 1/2
switch(config-if)# vrf member NewVRF
switch(config-if)# ip address 192.0.2.1/16
switch(config-if)# ip router isis Enterprise
switch(config-if)# copy running-config startup-config
```

Tuning IS-IS

You can tune IS-IS to match your network requirements.

You can use the following optional commands in router configuration mode to tune IS-IS:

Command	Purpose
lsp-gen-interval [**level-1** | **level-2**] *lsp-max-wait* [*lsp-initial-wait* *lsp-second-wait*] **Example** : `switch(config-router)# lsp-gen-interval level-1 500 500 500`	Configures the IS-IS throttle for LSP generation. The optional parameters are as follows: • lsp-max-wait—The maximum wait between the trigger and LSP generation. The range is from 500 to 65535 milliseconds. • lsp-initial-wait—The initial wait between the trigger and LSP generation. The range is from 50 to 65535 milliseconds. • lsp-second-wait—The second wait used for LSP throttle during backoff. The range is from 50 to 65535 milliseconds.
max-lsp-lifetime *lifetime* **Example:** `switch(config-router)# max-lsp-lifetime 500`	Sets the maximum LSP lifetime in seconds. The range is from 1 to 65535. The default is 1200.
metric-style transition **Example:** `switch(config-router)# metric-style transition`	Enables IS-IS to generate and accept both narrow metric-style Type Length Value (TLV) objects and wide metric-style TLV objects. The default is disabled.
spf-interval [**level-1** | **level-2**] *spf-max-wait* [*spf-initial-wait* *spf-second-wait*] **Example** : `switch(config-router)# spf-interval level-2 500 500 500`	Configures the interval between LSA arrivals. The optional parameters are as follows: • lsp-max-wait—The maximum wait between the trigger and SPF computation. The range is from 500 to 65535 milliseconds. • lsp-initial-wait—The initial wait between the trigger and SPF computation. The range is from 50 to 65535 milliseconds. • lsp-second-wait—The second wait used for SPF computation during backoff. The range is from 50 to 65535 milliseconds.

You can use the following optional command in router address configuration mode:

Command	Purpose
adjacency-check **Example** : `switch(config-router-af)# adjacency-check`	Performs an adjacency check to verify that an IS-IS instance forms an adjacency only with a remote IS-IS entity that supports the same address family. This command is enabled by default.

You can use the following optional commands in interface configuration mode to tune IS-IS:

Command	Purpose
isis csnp-interval *seconds* [**level-1** | **level-2**] **Example** : `switch(config-if)# isis csnp-interval 20`	Sets the complete sequence number PDU (CNSP) interval in seconds for IS-IS. The range is from 1 to 65535. The default is 10.
isis hello-interval *seconds* [**level-1** | **level-2**] **Example** : `switch(config-if)# isis hello-interval 20`	Sets the hello interval in seconds for IS-IS. The range is from 1 to 65535. The default is 10.
isis hello-multiplier num [**level-1** | **level-2**] **Example** : `switch(config-if)# isis hello-multiplier 20`	Specifies the number of IS-IS hello packets that a neighbor must miss before the router tears down an adjacency. The range is from 3 to 1000. The default is 3 .
isis lsp-interval milliseconds **Example:** `switch(config-if)# isis lsp-interval 20`	Sets the interval in milliseconds between LSPs sent on this interface during flooding. The range is from 10 to 65535. The default is 33.

Verifying the IS-IS Configuration

To display the IS-IS configuration, perform one of the following tasks:

Command	Purpose
show isis [*instance-tag*] **adjacency** [**interface**] [**detail** | **summary**] [**vrf** *vrf-name*]	Displays the IS-IS adjacencies. Use the **clear isis adjacency** command to clear these statistics. **Note** If the hostname is less than 14 characters, the **show isis adjacency** command displays the hostname. Otherwise, the System ID is displayed.
show isis [*instance-tag*] **database** [**level-1** | **level-2**] [**detail** | **summary**] [**LSP ID**] [{ **ip** | **ipv6** } **prefix ip-prefix**] | | [**router-id** *router-id*] | [**adjacency node-id**] | [**zero-sequence**]} [**vrf** *vrf-name*]	Displays the IS-IS LSP database.
show isis [*instance-tag*] **hostname** [**vrf** *vrf-name*]	Displays the dynamic host exchange information.
show isis [*instance-tag*] **interface** [**brief** | **interface**] [**level-1** | **level-2**] [**vrf** *vrf-name*]	Displays the IS-IS interface information.
show isis [*instance-tag*] **mesh-group** [*mesh-id*] [**vrf** *vrf-name*]	Displays the mesh group information.
show isis [*instance-tag*] **protocol** [**vrf** *vrf-name*]	Displays information about the IS-IS protocol.

Command	Purpose
show isis [*instance-tag*] { **ip** | **ipv6** } **redistribute route** [**ip-address** | **summary**] [[*ip-prefix*] [**longer-prefixes** [**summary**]] [**vrf** *vrf-name*]	Displays the IS-IS route redistribution information.
show isis [*instance-tag*] {**ip** | **ipv6** } **route** [**ip-address** | **summary**] [*ip-prefix* [**longer-prefixes** [**summary**]] [**detail**] [**vrf** *vrf-name*]	Displays the IS-IS route table.
show isis [*instance-tag*] **rrm** [*interface*] [**vrf** *vrf-name*]	Displays the IS-IS interface retransmission information.
show isis [*instance-tag*] **srm** [*interface*] [**vrf** *vrf-name*]	Displays the IS-IS interface flooding information.
show isis [*instance-tag*] **ssn** [*interface*] [**vrf** *vrf-name*]	Displays the IS-IS interface PSNP information.
show isis [*instance-tag*] {**ip** | **ipv6** } **summary-address** [*ip-address*] | [*ip-prefix*] [**vrf** *vrf-name*]	Displays the IS-IS summary address information.
show running-configuration isis	Displays the current running IS-IS configuration.
show tech-support isis [**detail**]	Displays the technical support details for IS-IS.

Monitoring IS-IS

To display IS-IS statistics, use the following commands:

Command	Purpose
show isis [*instance-tag*] **adjacency** [**interface**] [**system-ID**] [**detail**] [**summary**] [**vrf** *vrf-name*]	Displays the IS-IS adjacency statistics.
show isis [*instance-tag*] **database** [**level-1** | **level-2**] [**detail** | **summary**] [*lsip*] {[**adjacency** *id*] { **ip** | **ipv6** } **prefix** *prefix*] [**router-id** *id*] [**zero-sequence**]} [**vrf** *vrf-name*]	Displays the IS-IS database statistics.
show isis [*instance-tag*] **statistics** [*interface*] [**vrf** *vrf-name*]	Displays the IS-IS interface statistics.
show isis { **ip** | **ipv6** } **route-map statistics redistribute** {**bgp** *id* | **eigrp** *id* | **isis** *id* | **ospf** *id* | **rip** *id* | **static**} [**vrf** *vrf-name*]	Displays the IS-IS redistribution statistics.
show isis route-map statistics distribute { **level-1** | **level-2**} into { **level-1** | **level-2**}} [**vrf** *vrf-name*]	Displays IS-IS distribution statistics for routes distributed between levels.
show isis [*instance-tag*] **spf-log** [**detail**] [**vrf** *vrf-name*]	Displays the IS-IS SPF calculation statistics.
show isis [*instance-tag*] **traffic** [*interface*] [**vrf** *vrf-name*]	Displays the IS-IS traffic statistics.

To clear IS-IS configuration statistics, perform one of the following tasks:

Command	Purpose
clear isis [*instance-tag*] **adjacency** [***** | [*interface*] [**system-id** *id*]] [**vrf** *vrf-name*]	Clears the IS-IS adjacency statistics.
clear isis { **ip** | **ipv6** } **route-map statistics redistribute** {**bgp** *id* | **direct** | **eigrp** *id* | **isis** *id* | **ospf** *id* | **rip** *id* | **static**} [**vrf** *vrf-name*]	Clears the IS-IS redistribution statistics.
clear isis route-map statistics distribute { **level-1** | **level-2**} **into** { **level-1** | **level-2**} [**vrf** *vrf-name*]	Clears IS-IS distribution statistics for routes distributed between levels.
clear isis [*instance-tag*] **statistics** [***** | **interface**] [**vrf** *vrf-name*]	Clears the IS-IS interface statistics.
clear isis [*instance-tag*] **traffic** [***** | *interface*] [**vrf** *vrf-name*]	Clears the IS-IS traffic statistics.

Configuration Examples for IS-IS

This example shows how to configure IS-IS:

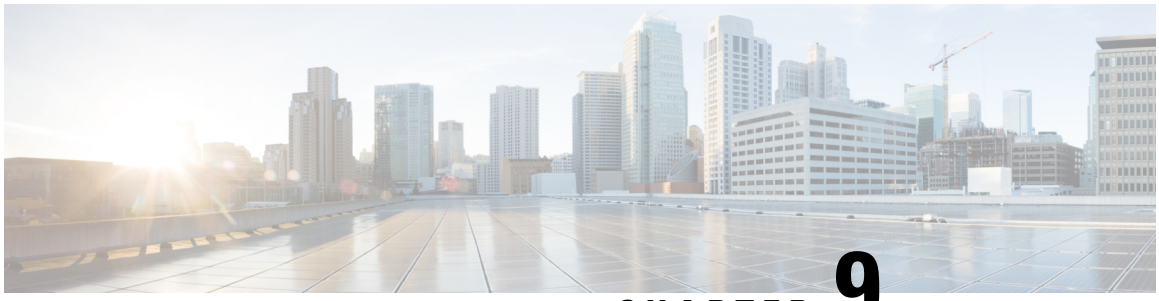
```
router isis Enterprise
is-type level-1
net 49.0001.0000.0000.0003.00
graceful-restart
address-family ipv4 unicast
default-information originate

interface ethernet 2/1
ip address 192.0.2.1/24
isis circuit-type level-1
ip router isis Enterprise
```

Related Topics

See the Configuring Route Policy Manager for more information on route maps.

Related Topics



CHAPTER 9

Configuring Basic BGP

This chapter describes how to configure Border Gateway Protocol (BGP) on Cisco NX-OS switches.

This chapter includes the following sections:

- About Basic BGP, on page 193
- Prerequisites for Basic BGP, on page 199
- Guidelines and Limitations for BGP, on page 199
- CLI Configuration Modes, on page 200
- Default Settings, on page 201
- Configuring Basic BGP, on page 202
- Verifying the Basic BGP Configuration, on page 213
- Displaying BGP Statistics, on page 215
- Configuration Examples for Basic BGP, on page 215
- Related Topics, on page 215
- Where to Go Next, on page 216
- Additional References, on page 216

About Basic BGP

Cisco NX-OS supports BGP version 4, which includes multiprotocol extensions that allow BGP to carry routing information for IP multicast routes and multiple Layer 3 protocol address families. BGP uses TCP as a reliable transport protocol to create TCP sessions with other BGP-enabled switches.

BGP uses a path-vector routing algorithm to exchange routing information between BGP-enabled networking switches or BGP speakers. Based on this information, each BGP speaker determines a path to reach a particular destination while detecting and avoiding paths with routing loops. The routing information includes the actual route prefix for a destination, the path of autonomous systems to the destination, and additional path attributes.

BGP selects a single path, by default, as the best path to a destination host or network. Each path carries well-known mandatory, well-known discretionary, and optional transitive attributes that are used in BGP best-path analysis. You can influence BGP path selection by altering some of these attributes by configuring BGP policies. See the Route Policies and Resetting BGP Sessions section for more information.

BGP also supports load balancing or equal-cost multipath (ECMP). See the Load Sharing and Multipath section for more information.

BGP Autonomous Systems

An autonomous system (AS) is a network controlled by a single administration entity. An autonomous system forms a routing domain with one or more interior gateway protocols (IGPs) and a consistent set of routing policies. BGP supports 16-bit and 32-bit autonomous system numbers. For more information, see the Autonomous Systems section.

Separate BGP autonomous systems dynamically exchange routing information through external BGP (eBGP) peering sessions. BGP speakers within the same autonomous system can exchange routing information through internal BGP (iBGP) peering sessions.

4-Byte AS Number Support

BGP supports 2-byte or 4-byte AS numbers. Cisco NX-OS displays 4-byte AS numbers in plain-text notation (that is, as 32-bit integers). You can configure 4-byte AS numbers as either plain-text notation (for example, 1 to 4294967295), or AS.dot notation (for example, 1.0). For more information, see the Autonomous Systems section.

Administrative Distance

An administrative distance is a rating of the trustworthiness of a routing information source. By default, BGP uses the administrative distances shown in the following table .

Table 15: BGP Default Administrative Distances

Distance	Default Value	Function
External	20	Applied to routes learned from eBGP.
Internal	200	Applied to routes learned from iBGP.
Local	200	Applied to routes originated by the router.

Note The administrative distance does not influence the BGP path selection algorithm, but it does influence whether BGP-learned routes are installed in the IP routing table.

For more information, see the Administrative Distance section.

BGP Peers

A BGP speaker does not discover another BGP speaker automatically. You must configure the relationships between BGP speakers. A BGP peer is a BGP speaker that has an active TCP connection to another BGP speaker.

BGP Sessions

BGP uses TCP port 179 to create a TCP session with a peer. When a TCP connection is established between peers, each BGP peer initially exchanges all of its routes—the complete BGP routing table—with the other

peer. After this initial exchange, the BGP peers send only incremental updates when a topology change occurs in the network or when a routing policy change occurs. In the periods of inactivity between these updates, peers exchange special messages called keepalives. The hold time is the maximum time limit that can elapse between receiving consecutive BGP update or keepalive messages.

Cisco NX-OS supports the following peer configuration options:

- Individual IPv4 or IPv6 address—BGP establishes a session with the BGP speaker that matches the remote address and AS number.
- IPv4 or IPv6 prefix peers for a single AS number—BGP establishes sessions with BGP speakers that match the prefix and the AS number.
- Dynamic AS number prefix peers—BGP establishes sessions with BGP speakers that match the prefix and an AS number from a list of configured AS numbers.

Dynamic AS Numbers for Prefix Peers

Cisco NX-OS accepts a range or list of AS numbers to establish BGP sessions. For example, if you configure BGP to use IPv4 prefix 192.0.2.0/8 and AS numbers 33, 66, and 99, BGP establishes a session with 192.0.2.1 with AS number 66 but rejects a session from 192.0.2.2 with AS number 50.)

Cisco NX-OS does not associate prefix peers with dynamic AS numbers as either interior BGP (iBGP) or external BGP (eBGP) sessions until after the session is established. See Configuring Advanced BGP, for more information on iBGP and eBGP.

Note The dynamic AS number prefix peer configuration overrides the individual AS number configuration that is inherited from a BGP template. See Configuring Advanced BGP, for more information on templates.

BGP Router Identifier

To establish BGP sessions between peers, BGP must have a router ID , which is sent to BGP peers in the OPEN message when a BGP session is established. The BGP router ID is a 32-bit value that is often represented by an IPv4 address. You can configure the router ID. By default, Cisco NX-OS sets the router ID to the IPv4 address of a loopback interface on the router. If no loopback interface is configured on the router, then the software chooses the highest IPv4 address configured to a physical interface on the router to represent the BGP router ID. The BGP router ID must be unique to the BGP peers in a network.

If BGP does not have a router ID, it cannot establish any peering sessions with BGP peers.

BGP Path Selection

Although BGP might receive advertisements for the same route from multiple sources, BGP selects only one path as the best path. BGP puts the selected path in the IP routing table and propagates the path to its peers.

The best-path algorithm runs each time that a path is added or withdrawn for a given network. The best-path algorithm also runs if you change the BGP configuration. BGP selects the best path from the set of valid paths available for a given network.

Cisco NX-OS implements the BGP best-path algorithm in the following steps:

Procedure

Step 1 Compares two paths to determine which is better (see the Step 1—Step 1—Comparing Pairs of Paths section).

Step 2 Iterates over all paths and determines in which order to compare the paths to select the overall best path (see the Step 2—Step 2—Determining the Order of Comparisons section).

Step 3 Determines whether the old and new best paths differ enough so that the new best path should be used (see the Step 3—Step 3—Determining the Best-Path Change Suppression section).

Example

Note The order of comparison determined in Part 2 is important. Consider the case where you have three paths, A, B, and C. When Cisco NX-OS compares A and B, it chooses A. When Cisco NX-OS compares B and C, it chooses B. But when Cisco NX-OS compares A and C, it might not choose A because some BGP metrics apply only among paths from the same neighboring autonomous system and not among all paths.

The path selection uses the BGP AS-path attribute. The AS-path attribute includes the list of autonomous system numbers (AS numbers) traversed in the advertised path. If you subdivide your BGP autonomous system into a collection or confederation of autonomous systems, the AS path contains confederation segments that list these locally defined autonomous systems.

Step 1—Comparing Pairs of Paths

This first step in the BGP best-path algorithm compares two paths to determine which path is better. The following sequence describes the basic steps that Cisco NX-OS uses to compare two paths to determine the better path:

1. Cisco NX-OS chooses a valid path for comparison. (For example, a path that has an unreachable next-hop is not valid.)
2. Cisco NX-OS chooses the path with the highest weight.
3. Cisco NX-OS chooses the path with the highest local preference.
4. If one of the paths is locally originated, Cisco NX-OS chooses that path.
5. Cisco NX-OS chooses the path with the shorter AS path.

Note When calculating the length of the AS path, Cisco NX-OS ignores confederation segments, and counts AS sets as 1. See the AS Confederations section for more information.

6. Cisco NX-OS chooses the path with the lower origin. The Interior Gateway Protocol (IGP) is considered lower than EGP.
7. 7. Cisco NX-OS chooses the path with the lower multi-exit discriminator (MED).

You can configure a number of options that affect whether or not this step is performed. In general, Cisco NX-OS compares the MED of both paths if the paths were received from peers in the same autonomous system; otherwise, Cisco NX-OS skips the MED comparison.

You can configure Cisco NX-OS to always perform the best-path algorithm MED comparison, regardless of the peer autonomous system in the paths. See the Tuning the Best-Path Algorithm section for more information. Otherwise, Cisco NX-OS will perform a MED comparison that depends on the AS-path attributes of the two paths being compared:

a. If a path has no AS path or the AS path starts with an AS_SET, then the path is internal, and Cisco NX-OS compares the MED to other internal paths.

b. If the AS path starts with an AS_SEQUENCE, then the peer autonomous system is the first AS number in the sequence, and Cisco NX-OS compares the MED to other paths that have the same peer autonomous system.

c. If the AS path contains only confederation segments or starts with confederation segments followed by an AS_SET, the path is internal and Cisco NX-OS compares the MED to other internal paths.

d. If the AS path starts with confederation segments followed by an AS_SEQUENCE, then the peer autonomous system is the first AS number in the AS_SEQUENCE, and Cisco NX-OS compares the MED to other paths that have the same peer autonomous system.

Note If Cisco NX-OS receives no MED attribute with the path, then Cisco NX-OS considers the MED to be 0 unless you configure the best-path algorithm to set a missing MED to the highest possible value. See the Tuning the Best-Path Algorithm section for more information.

e. If the nondeterministic MED comparison feature is enabled, the best path algorithm uses the Cisco IOS style of MED comparison. See the Tuning the Best-Path Algorithm section for more information.

8. 8. If one path is from an internal peer and the other path is from an external peer, then Cisco NX-OS chooses the path from the external peer.

9. If the paths have different IGP metrics to their next-hop addresses, then Cisco NX-OS chooses the path with the lower IGP metric.

10. Cisco NX-OS uses the path that was selected by the best-path algorithm the last time that it was run.

    If all path parameters in Step 1 through Step 9 are the same, then you can configure the best-path algorithm to compare the router IDs. See the Tuning the Best-Path Algorithm section for more information. If the path includes an originator attribute, then Cisco NX-OS uses that attribute as the router ID to compare to; otherwise, Cisco NX-OS uses the router ID of the peer that sent the path. If the paths have different router IDs, Cisco NX-OS chooses the path with the lower router ID.

    When using the attribute originator as the router ID, it is possible that two paths have the same router ID. It is also possible to have two BGP sessions with the same peer router, and therefore you can receive two paths with the same router ID.

11. Cisco NX-OS selects the path with the shorter cluster length. If a path was not received with a cluster list attribute, the cluster length is 0.

12. Cisco NX-OS chooses the path received from the peer with the lower IP address. Locally generated paths (for example, redistributed paths) have a peer IP address of 0.

Paths that are equal after step 9 can be used for multipath if you configure multipath. See the Load Sharing and Multipathsection for more information.

Step 2—Determining the Order of Comparisons

The second step of the BGP best-path algorithm implementation is to determine the order in which Cisco NX-OS compares the paths:

1. Cisco NX-OS partitions the paths into groups. Within each group Cisco NX-OS compares the MED among all paths. Cisco NX-OS uses the same rules as in the Step 1—Comparing Pairs of Paths section to determine whether MED can be compared between any two paths. Typically, this comparison results in one group being chosen for each neighbor autonomous system. If you configure the **bgp bestpath med always** command, then Cisco NX-OS chooses just one group that contains all the paths.
2. Cisco NX-OS determines the best path in each group by iterating through all paths in the group and keeping track of the best one so far. Cisco NX-OS compares each path with the temporary best path found so far and if the new path is better, it becomes the new temporary best path and Cisco NX-OS compares it with the next path in the group.
3. Cisco NX-OS forms a set of paths that contain the best path selected from each group in Step 2. Cisco NX-OS selects the overall best path from this set of paths by going through them as in Step 2.

Step 3—Determining the Best-Path Change Suppression

The next part of the implementation is to determine whether Cisco NX-OS will use the new best path or suppress the new best path. The router can continue to use the existing best path if the new one is identical to the old path (if the router ID is the same). Cisco NX-OS continues to use the existing best path to avoid route changes in the network.

You can turn off the suppression feature by configuring the best-path algorithm to compare the router IDs. See the Tuning the Best-Path Algorithm section for more information. If you configure this feature, the new best path is always preferred to the existing one.

You cannot suppress the best-path change if any of the following conditions occur:

- The existing best path is no longer valid.
- Either the existing or new best paths were received from internal (or confederation) peers or were locally generated (for example, by redistribution).
- The paths were received from the same peer (the paths have the same router ID).
- The paths have different weights, local preferences, origins, or IGP metrics to their next-hop addresses.
- The paths have different MEDs.

BGP and the Unicast RIB

BGP communicates with the unicast routing information base (unicast RIB) to store IPv4 routes in the unicast routing table. After selecting the best path, if BGP determines that the best path change needs to be reflected in the routing table, it sends a route update to the unicast RIB.

BGP receives route notifications regarding changes to its routes in the unicast RIB. It also receives route notifications about other protocol routes to support redistribution.

BGP also receives notifications from the unicast RIB regarding next-hop changes. BGP uses these notifications to keep track of the reachability and IGP metric to the next-hop addresses.

Whenever the next-hop reachability or IGP metrics in the unicast RIB change, BGP triggers a best-path recalculation for affected routes.

BGP Virtualization

BGP supports virtual routing and forwarding (VRF) instances. By default, Cisco NX-OS places you in the default VRF unless you specifically configure another VRF. For more information, see Configuring Layer 3 Virtualization.

Prerequisites for Basic BGP

BGP has the following prerequisites:

- You must enable the BGP feature (see the Enabling the BGP Feature section).
- You should have a valid router ID configured on the system.
- You must have an AS number, either assigned by a Regional Internet Registry (RIR) or locally administered.
- You must configure at least one IGP that is capable of recursive next-hop resolution.
- You must configure an address family under a neighbor for the BGP session establishment.

Guidelines and Limitations for BGP

BGP has the following configuration guidelines and limitations:

- The dynamic AS number prefix peer configuration the overrides individual AS number configuration inherited from a BGP template.
- If you configure a dynamic AS number for prefix peers in an AS confederation, BGP establishes sessions with only the AS numbers in the local confederation.
- BGP sessions created through a dynamic AS number prefix peer ignore any configured external BGP (eBGP) multihop time-to-live (TTL) value or a disabled check for directly connected peers.
- You must configure a router ID for BGP to avoid automatic router ID changes and session flaps.
- You must use the maximum-prefix configuration option per peer to restrict the number of routes received and system resources used.
- You must configure the update source to establish a session with BGP/eBGP multihop sessions.
- You must specify a BGP policy if you configure redistribution.
- You must define the BGP router ID within a VRF.
- If you decrease the keepalive and hold timer values, you might experience BGP session flaps.
- If you configure VRFs, enter the desired VRF (see Configuring Layer 3 Virtualization).

CLI Configuration Modes

The following sections describe how to enter each of the CLI configuration modes for BGP. From a mode, you can enter the **?** command to display the commands available in that mode.

Global Configuration Mode

Use global configuration mode to create a BGP process and configure advanced features such as AS confederation and route dampening. For more information, see Configuring Advanced BGP.

This example shows how to enter router configuration mode:

```
switch# configuration
switch(config)# router bgp 64496
switch(config-router)#
```

BGP supports virtual routing and forwarding (VRF). You can configure BGP within the appropriate VRF if you are using VRFs in your network. See the Configuring Virtualization section for more information.

This example shows how to enter VRF configuration mode:

```
switch(config)# router bgp 64497
switch(config-router)# vrf vrf_A
switch(config-router-vrf)#
```

Address Family Configuration Mode

You can optionally configure the address families that BGP supports. Use the **address-family** command in router configuration mode to configure features for an address family. Use the **address-family** command in neighbor configuration mode to configure the specific address family for the neighbor.

You must configure the address families if you are using route redistribution, address aggregation, load balancing, and other advanced features.

This example shows how to enter address family configuration mode from the router configuration mode:

```
switch(config)# router bgp 64496
switch(config-router)# address-family ipv6 unicast
switch(config-router-af)#
```

This example shows how to enter VRF address family configuration mode if you are using VRFs:

```
switch(config)# router bgp 64497
switch(config-router)# vrf vrf_A
switch(config-router-vrf)# address-family ipv6 unicast
switch(config-router-vrf-af)#
```

Neighbor Configuration Mode

Cisco NX-OS provides the neighbor configuration mode to configure BGP peers. You can use neighbor configuration mode to configure all parameters for a peer.

This example shows how to enter neighbor configuration mode:

```
switch(config)# router bgp 64496
switch(config-router)# neighbor 192.0.2.1
switch(config-router-neighbor)#
```

This example shows how to enter VRF neighbor configuration mode:

```
switch(config)# router bgp 64497
switch(config-router)# vrf vrf_A
switch(config-router-vrf)# neighbor 192.0.2.1
switch(config-router-vrf-neighbor)#
```

Neighbor Address Family Configuration Mode

An address family configuration submode inside the neighbor configuration submode is available for entering address family-specific neighbor configuration and enabling the address family for the neighbor. Use this mode for advanced features such as limiting the number of prefixes allowed for this neighbor and removing private AS numbers for eBGP.

With RFC 5549, you can configure an IPv4 address family for a neighbor with an IPv6 address.

This example shows how to enter the IPv4 neighbor address family configuration mode for a neighbor with an IPv4 address:

```
switch(config)# router bgp 64496
switch(config-router# neighbor 192.0.2.1
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)#
```

This example shows how to enter the IPv4 neighbor address family configuration mode for a neighbor with an IPv6 address:

```
switch(config)# router bgp 64496
switch(config-router# neighbor 2001:db8::/64 eui64
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)#
```

This example shows how to enter the VRF IPv4 neighbor address family configuration mode for a neighbor with an IPv4 address:

```
switch(config)# router bgp 64497
switch(config-router)# vrf vrf_A
switch(config-router-vrf)# neighbor 209.165.201.1
switch(config-router-vrf-neighbor)# address-family ipv4 unicast
switch(config-router-vrf-neighbor-af)#
```

This example shows how to enter the VRF IPv4 neighbor address family configuration mode for a neighbor with an IPv6 address:

```
switch(config)# router bgp 64497
switch(config-router)# vrf vrf_A
switch(config-router-vrf)# neighbor 2001:db8::/64 eui64
switch(config-router-vrf-neighbor)# address-family ipv4 unicast
switch(config-router-vrf-neighbor-af)#
```

Default Settings

Following table lists the default settings for BGP parameters.

Table 16: Default BGP Parameters

Parameters	Default
BGP feature	Disabled

Parameters	Default
Keep alive interval	60 seconds
Hold timer	180 seconds

Configuring Basic BGP

To configure a basic BGP, you need to enable BGP and configure a BGP peer. Configuring a basic BGP network consists of a few required tasks and many optional tasks. You must configure a BGP routing process and BGP peers.

Enabling the BGP Feature

Before you begin

You must enable the BGP feature before you can configure BGP.

Procedure

	Command or Action	Purpose

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

Step 2	**feature bgp** **Example:** `switch(config)# feature bgp`	Enables the BGP feature.
Step 3	(Optional) **show feature** **Example:** `switch(config)# show feature`	Displays enabled and disabled features.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

Use the **no feature bgp** command to disable the BGP feature and remove all associated configuration.

Command	Purpose
no feature bgp **Example:** `switch(config)# no feature bgp`	Disables the BGP feature and removes all associated configuration.

Creating a BGP Instance

You can create a BGP instance and assign a router ID to the BGP instance. See the BGP Router Identifier section. Cisco NX-OS supports 2-byte or 4-byte autonomous system (AS) numbers in plain-text notation or as.dot notation. See the 4-Byte AS Number Support section for more information.

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

BGP must be able to obtain a router ID (for example, a configured loopback address).

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *autonomous-system-number*
	Example:
	`switch(config)# router bgp 64496`
	`switch(config-router)#`
		Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 3	(Optional) **router-id** *ip-address* **Example:** `switch(config-router)# router-id 192.0.2.255`	Configures the BGP router ID. This IP address identifies this BGP speaker. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

Step 4	(Optional) **address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters global address family configuration mode for the specified address family. This command triggers an automatic notification and session reset for all BGP neighbors.

Step 5	(Optional) **network** *ip-prefix* [**route-map** *map-name*] **Example:**	Specifies a network as local to this autonomous system and adds it to the BGP routing table. For exterior protocols, the network command controls which networks are advertised. Interior

	Command or Action	Purpose
	`switch(config-router-af)# network 192.0.2.0`	protocols use the network command to determine where to send updates.
Step 6	(Optional) **show bgp all** **Example:** `switch(config-router-af)# show bgp all`	Displays information about all BGP address families.
Step 7	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-af)# copy running-config startup-config`	Saves this configuration change.

Example

Use the **no router bgp** command to remove the BGP process and the associated configuration.

Command	Purpose
no router bgp *autonomous-system-number* **Example:** `switch(config)# no router bgp 201`	Deletes the BGP process and the associated configuration.

This example shows how to enable BGP with the IPv4 unicast address family and manually add one network to advertise:

```
switch# configure terminal
switch(config)# router bgp 64496
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# network 192.0.2.0
switch(config-router-af)# copy running-config startup-config
```

Restarting a BGP Instance

You can restart a BGP instance and clear all peer sessions for the instance.

To restart a BGP instance and remove all associated peers, use the following command:

Command	Purpose
restart bgp *instance-tag* **Example:** `switch(config)# restart bgp 201`	Restarts the BGP instance and resets or reestablishes all peering sessions.

Shutting Down BGP

You can shut down the BGP protocol and gracefully disable BGP and retain the configuration.

To shut down BGP, use the following command in router configuration mode:

Command	Purpose
shutdown **Example:** `switch(config-router)# shutdown`	Gracefully shuts down BGP.

Configuring BGP Peers

You can configure a BGP peer within a BGP process. Each BGP peer has an associated keepalive timer and hold timers. You can set these timers either globally or for each BGP peer. A peer configuration overrides a global configuration.

Note You must configure the address family under neighbor configuration mode for each peer.

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *autonomous-system-number*
	Example:
	`switch(config)# router bgp 64496`
	`switch(config-router)#`
		Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 3	**neighbor** *ip-address* { **ipv4** | **ipv6** } **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 209.165.201.1 remote-as 64497`
	`switch(config-router-neighbor)#`
		Configures the specified address type and AS number for a remote BGP peer. The *ip-address* format is x.x.x.x. The IPv6 *address-format* is A:B::C:D.

Step 4	(Optional) **description** *text*
	Example:
	`switch(config-router-neighbor)# description Peer Router B`
	`switch(config-router-neighbor)#`
		Adds a description for the neighbor. The description is an alphanumeric string up to 80 characters.

	Command or Action	Purpose
Step 5	(Optional) **timers** *keepalive-time hold-time* **Example:** `switch(config-router-neighbor )# timers 30 90`	Adds the keepalive and hold time BGP timer values for the neighbor. The range is from 0 to 3600 seconds. The default is 60 seconds for the keepalive time and 180 seconds for the hold time. **Note** BGP sessions with a hold-timer of 10 seconds or less are not effective until the BGP session has been up for 60 seconds or more. Once the session has been up for 60 seconds, the hold-timer will work as configured.
Step 6	(Optional) **shutdown** **Example:** `switch(config-router-neighbor)# shutdown`	Administratively shuts down this BGP neighbor. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

Step 7	**address-family** { **ipv4** | **ipv6**} { **unicast** | **multicast**}
	Example:
	`switch(config-router-neighbor)# address-family ipv4 unicast`
	`switch(config-router-neighbor-af)#`
		Enters neighbor address family configuration mode for the unicast specified address family.

Step 8	(Optional) **show bgp** { **ipv4** | **ipv6**} { **unicast** | **multicast** } **neighbors** **Example:** `switch(config-router-neighbor-af)# show bgp ipv4 unicast neighbors`	Displays information about BGP peers.
Step 9	**copy running-config startup-config** **Example:** `switch(config-router-neighbor-af) copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure a BGP peer:

```
switch# configure terminal
switch(config)# router bgp 64496
switch(config-router)# neighbor 192.0.2.1 remote-as 64497
switch(config-router-neighbor)# description Peer Router B
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# copy running-config startup-config
```

Distributing the Default Static Route to All BGP VRFs

Ensure that you have enabled the BGP feature.

Before you begin

A new CLI default-information originate is added to distribute the default static route from non-default VRF to all BGP (VRF) Virtual Router Context and to install that route in the local BGP route table for all the BGP VRFs.

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *autonomous-system-number*
	Example:
	`switch(config)# router bgp 64496`
	`switch(config-router)#`
		Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 3	**vrf** *vrf-name* **Example:** `switch(config-router)# vrf vrf1`	Configures the VRF name.

Step 4	**address-family** { **ipv4** | **ipv6**} { **unicast** | **multicast** }
	Example:
	`switch(config-router-neighbor)#`
	`address-family ipv4 unicast`
	`switch(config-router-neighbor-af)#`
		Enters global address family configuration mode for the specified address family. This command triggers an automatic notification and session reset for all BGP neighbors.

Step 5	**redistribute static route-map map-name**
	Example:
	`switch(config-router-neighbor-af)#`
	`redistribute static route-map test`
		Redistributes the static route map.

Step 6	**default-information originate**
	Example:
	`switch(config-router-neighbor-af)#`
	`default-information originate`
		Distributes the default static route from non-default VRF to all BGP (VRF) Virtual Router Context and to install that route in the local BGP route table for all the BGP VRFs.

Step 7	**copy running-config startup-config**
	Example:
	`switch(config-router-neighbor-af)# copy`
	`running-config startup-config`
		Saves this configuration change.

Example

This example shows how to distribute the default static route from non-default VRF to all BGP (VRF) Virtual Router Context and to install that route in the local BGP route table for all the BGP VRFs:

```
switch# configure terminal
switch(config)# router bgp 100
switch(config-router)# vrf green
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-vrf-af)# redistribute static route-map test
switch(config-router-vrf-af)# d efault-information originate
switch(config-router-vrf-af)# exit
switch(config)# copy running-config startup-confi g
```

This example shows how to configure default-route leaking from non-default VRF Green to another non-default VRF Shared:

```
router bgp 100
address-family ipv4 unicast
redistribute static route-map test
vrf Green
address-family ipv4 unicast
redistribute static route-map test
default-information originate
vrf Shared
address-family ipv4 unicast
redistribute static route-map test
```

Configuring Update Announcement Delay Timers

You can configure a minimum route advertisement interval (MRAI) between the sending of BGP routing updates by using the advertisement-interval command. Instead of announcing route updates as soon as they are triggered, BGP waits for the advertisement interval to expire before sending out the update. If there are other changes that happen during this interval, BGP can announce all these changes efficiently.

This delay timer is configured only for new route announcements and not for route withdrawal announcements. Route withdrawals must be announced immediately.

Before you begin

Ensure that you have enabled the BGP feature.

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *autonomous-system-number*
	Example:
	`switch(config)# router bgp 64496`
	`switch(config-router)#`
		Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of

	Command or Action	Purpose
		a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 3	**neighbor** *prefix* **remote-as route-map** *map-name*
	Example:
	`switch(config-router)# neighbor 192.0.2.0/8 remote-as routemap BGPPeers`
	`switch(config-router-neighbor)#`
		Configures the IPv4 prefix and a route map for the list of accepted AS numbers for the remote BGP peers. The *prefix* format for IPv4 is x.x.x.x/length. The length range is from 1 to 32.
		The *map-name* can be any case-sensitive, alphanumerics string up to 63 characters.

Step 4	**address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }
	Example:
	`switch(config-router-neighbor)# address-family ipv4 unicast`
	`switch(config-router-neighbor-af)#`
		Enters global address family configuration mode for the specified address family. This command triggers an automatic notification and session reset for all BGP neighbors.

Step 5	**advertisement-interval seconds** **Example:** `switch(config-router-neighbor-af)# advertisement-interval 300`	Configures an interval that delays the announcement of new route updates. **Note** This interval does not apply to route withdrawal announcements, which must be made immediately. The interval can range between 1 and 600 seconds.

Example

This example shows how to configure an update announcement delay timer:

```
switch# configure terminal
switch(config)# router bgp 64496
switch(config-router)# neighbor 192.0.2.0/8 remote-as route-map BGPPeers
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# advertisement-interval 300
```

Configuring BGP Reconnect Interval

You can configure an interval after which a BGP session can reconnect.

Before you begin

Ensure that you have enabled the BGP feature.

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *autonomous-system-number*
	Example:
	`switch(config)# router bgp 64496`
	`switch(config-router)#`
		Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 3	**reconnect-interval** *interval* **Example:** `switch(config-router)# reconnect-interval 20`	Configures the interval after which a dropped BGP connection can automatically reconnect. The *interval* range is from 1 second to 60 seconds. The default value of the interval is 30 seconds.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure the BGP reconnect interval:

```
switch# configure terminal
switch(config)# router bgp 64496
switch(config-router)# reconnect-interval 20
switch(config-router)# copy running-config startup-config
```

Configuring Dynamic AS Numbers for Prefix Peers

You can configure multiple BGP peers within a BGP process. You can limit BGP session establishment to a single AS number or multiple AS numbers in a route map.

BGP sessions configured through dynamic AS numbers for prefix peers ignore the **ebgp-multihop** command and the **disable-connected-check** command.

You can change the list of AS numbers in the route map, but you must use the **no neighbor** command to change the route-map name. Changes to the AS numbers in the configured route map affect only new sessions.

Before you begin

Ensure that you have enabled the BGP feature.

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *autonomous-system-number*
	Example:
	`switch(config)# router bgp 64496`
	`switch(config-router)#`
		Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 3	**neighbor** *prefix* **remote-as route-map** *map-name*
	Example:
	`switch(config-router)# neighbor 192.0.2.0/8 remote-as routemap BGPPeers`
	`switch(config-router-neighbor)#`
		Configures the IPv4 prefix and a route map for the list of accepted AS numbers for the remote BGP peers. The *prefix* format for IPv4 is x.x.x.x/length. The length range is from 1 to 32.
		The *map-name* can be any case-sensitive, alphanumerics string up to 63 characters.

Step 4	(Optional) **show bgp** { **ipv4** { **unicast** | **multicast** } **neighbors** **Example:** `switch(config-router-neighbor)# show bgp ipv4 unicast neighbors`	Displays information about BGP peers.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-neighbor) copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure dynamic AS numbers for a prefix peer:

```
switch# configure terminal
switch(config)# route-map BGPPeers
switch(config-route-map)# match as-number 64496, 64501-64510
switch(config-route-map)# match as-number as-path-list List1, List2
switch(config-route-map)# exit
switch(config)# router bgp 64496
switch(config-router)# neighbor 192.0.2.0/8 remote-as route-map BGPPeers
switch(config-router-neighbor)# description Peer Router B
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# copy running-config startup-config
```

Clearing BGP Information

To clear BGP information, use the following commands:

Command	Purpose
clear bgp all { *neighbor* | * | *as-number* | **peer-template** *name* | *prefix* } [**vrf** *vrf-name*]	Clears one or more neighbors from all address families. * clears all neighbors in all address families. The arguments are as follows: • *neighbor* —IPv4 address of a neighbor. • *as-number* —Autonomous system number. The AS number can be a 16-bit integer or a 32-bit integer in the form of higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format • *name* —Peer template name. The name can be any case-sensitive, alphanumeric string up to 64 characters. • *prefix* —IPv4 prefix. All neighbors within that prefix are cleared • *vrf-name* —VRF name. All neighbors in that VRF are cleared. The name can be any case-sensitive, alphanumeric string up to 64 characters.
clear bgp all dampening [**vrf** *vrf-name*]	Clears route flap dampening networks in all address families. The *vrf-name* can be any case-sensitive, alphanumeric string up to 64 characters.
clear bgp all flap-statistics [**vrf** *vrf-name*]	Clears route flap statistics in all address families. The *vrf-name*can be any case-sensitive, alphanumeric string up to 64 characters.
clear bgp ipv4 { **unicast** | **multicast** } **dampening** [**vrf** *vrf-name*]	Clears route flap dampening networks in the selected address family. The *vrf-name*can be any case-sensitive, alphanumeric string up to 64 characters.
clear bgp ipv4 { **unicast** | **multicast** } **flap-statistics** [**vrf** *vrf-name*]	Clears route flap statistics in the selected address family. The *vrf-name*can be any case-sensitive, alphanumeric string up to 64 characters.
clear bgp { **ipv4** | **ipv6** }{ **unicast** | **multicast** } { *neighbor* | * | *as-number* | **peer-template** *name* | *prefix* } [**vrf** *vrf-name*]	Clears one or more neighbors from the selected address family. * clears all neighbors in the address family. The arguments are as follows: • neighbor —IPv4 address of a neighbor. • as-number — Autonomous system number. The AS number can be a 16-bit integer or a 32-bit integer in the form of higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format. • name —Peer template name. The name can be any case-sensitive, alphanumeric string up to 64 characters. • prefix —IPv4 prefix. All neighbors within that prefix are cleared. • vrf-name —VRF name. All neighbors in that VRF are cleared. The name can be any case-sensitive, alphanumeric string up to 64 characters

Command	Purpose
clear ip bgp { **ip** { **unicast** | **multicast** }} { *neighbor* | * | *as-number* | **peer-template** *name* | *prefix* } [**vrf** *vrf-name*]	Clears one or more neighbors. * clears all neighbors in the address family. The arguments are as follows: • neighbor —IPv4 address of a neighbor. • as-number — Autonomous system number. The AS number can be a 16-bit integer or a 32-bit integer in the form of higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format. • name —Peer template name. The name can be any case-sensitive, alphanumeric string up to 64 characters. • prefix —IPv4 prefix. All neighbors within that prefix are cleared. • vrf-name —VRF name. All neighbors in that VRF are cleared. The name can be any case-sensitive, alphanumeric string up to 64 characters.
clear ip bgp dampening [*ip-neighbor* | *ip-prefix*] [**vrf** *vrf-name*]	Clears route flap dampening in one or more networks. The arguments are as follows: • ip-neighbor —IPv4 or IPv6 address of a neighbor. • ip-prefix —IPv4 or IPv6. All neighbors within that prefix are cleared. • vrf-name —VRF name. All neighbors in that VRF are cleared. The name can be any case-sensitive, alphanumeric string up to 64 characters.
clear ip bgp flap-statistics [*ip-neighbor* | *ip-prefix*] [**vrf** *vrf-name*]	Clears route flap statistics in one or more networks. The arguments are as follows: • ip-neighbor —IPv4 or IPv6 address of a neighbor. • ip-prefix —IPv4 or IPv6. All neighbors within that prefix are cleared. • vrf-name —VRF name. All neighbors in that VRF are cleared. The name can be any case-sensitive, alphanumeric string up to 64 characters.

Verifying the Basic BGP Configuration

To display the BGP configuration information, perform the following tasks:

Command	Purpose
show bgp all [**summary**] [**vrf** *vrf-name*]	Displays the BGP information for all address families.
show bgp convergence [**vrf** *vrf-name*]	Displays the BGP information for all address families.

Command	Purpose
show bgp { **ipv4** { **unicast** | **multicast** } [**ip-address**] **community** { **regexp** *expression* | [**community**] [**no-advertise**] [**no-export**] [**no-export-subconfed**]} [**vrf vrf-name**]	Displays the BGP routes that match a BGP community.
show bgp [**vrf vrf-name**] { **ip** | **ipv6**} {**unicast** | **multicast**} [**ip-address**] **community-list** *list-name* [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP community list.
show bgp ip { **unicast** | **multicast** } [**ip-address**] **extcommunity** {**regexp** *expression* | **generic** [**non-transitive** | **transitive**] *aa4:nn* [**exact-match**]} [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP extended community.
show bgp ip { **unicast** | **multicast** } [**ip-address**] **extcommunity-list** *list-name* [**exact-match**] [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP extended community list.
show bgp ip { **unicast** | **multicast** } [**ip-address**] {**dampening** *dampened-paths* [**regexp** *expression*]} [**vrf** *vrf-name*]	Displays the information for BGP route dampening. Use the **clear bgp dampening** command to clear the route flap dampening information.
show bgp ip { **unicast** | **multicast** } [**ip-address**] **history-paths** [**regexp** *expression*] [**vrf** *vrf-name*]	Displays the BGP route history paths.
show bgp ip { **unicast** | **multicast** } [**ip-address**] **filter-list list-name** [**vrf** vrf-name]	Displays the information for the BGP filter list.
show bgp ip { **unicast** | **multicast** } [**ip-address**] **neighbors** [**ip-address**] [**vrf** *vrf-name*]	Displays the information for BGP peers. Use the **clear bgp neighbors** command to clear these neighbors.
show bgp ip { **unicast** | **multicast** } [**ip-address**] { **nexthop** | **nexthop-database**} [**vrf** *vrf-name*]	Displays the information for the BGP route next-hop.
show bgp paths	Displays the BGP path information.
show bgp ip { **unicast** | **multicast** } [**ip-address**] **policy name** [**vrf** *vrf-name*]	Displays the BGP policy information. Use the **clear bgp policy** command to clear the policy information.
show bgp ip { **unicast** | **multicast** } [**ip-address**] **prefix-list** *list-name* [**vrf** *vrf-name*]	Displays the BGP routes that match the prefix list.
show bgp ip { **unicast** | **multicast** } [**ip-address**] **received-paths** [**vrf** *vrf-name*]	Displays the BGP paths stored for soft reconfiguration.
show bgp ip { **unicast** | **multicast** } [**ip-address**] **regexp** *expression* [**vrf** *vrf-name*]	Displays the BGP routes that match the AS_path regular expression.
show bgp ip { **unicast** | **multicast** } [**ip-address**] **route-map** *map-name* [**vrf** *vrf-name*]	Displays the BGP routes that match the route map.

Command	Purpose
show bgp peer-policy name [**vrf** *vrf-name*]	Displays the information about BGP peer policies.
show bgp peer-session name [**vrf** *vrf-name*]	Displays the information about BGP peer sessions.
show bgp peer-template name [**vrf** *vrf-name*]	Displays the information about BGP peer templates. Use the **clear bgp peer-template** command to clear all neighbors in a peer template.
show bgp process	Displays the BGP process information.
show ip bgp *options*	Displays the BGP status and configuration information. This command has multiple options.
show running-configuration bgp	Displays the current running BGP configuration.

Displaying BGP Statistics

To display BGP statistics, use the following commands:

Command	Purpose
show bgp ip { **unicast** | **multicast** } [**ip-address**] **flap-statistics** [**vrf** *vrf-name*]	Displays the BGP route flap statistics. Use the **clear bgp flap-statistics** command to clear these statistics.
show bgp sessions [**vrf** *vrf-name*]	Displays the BGP sessions for all peers. Use the **clear bgp sessions** command to clear these statistics.
show bgp sessions [**vrf** *vrf-name*]	Displays the BGP sessions for all peers. Use the **clear bgp sessions** command to clear these statistics.
show bgp statistics	Displays the BGP statistics.

Configuration Examples for Basic BGP

This example shows a basic BGP configuration:

```
feature bgp
router bgp 64496
neighbor 192.0.2.1 remote-as 64496
address-family ipv4 unicast
next-hop-self
```

Related Topics

The following topics relate to BGP:

- Configuring Route Policy Manager

Where to Go Next

See Configuring Advanced BGP for details on the following features:

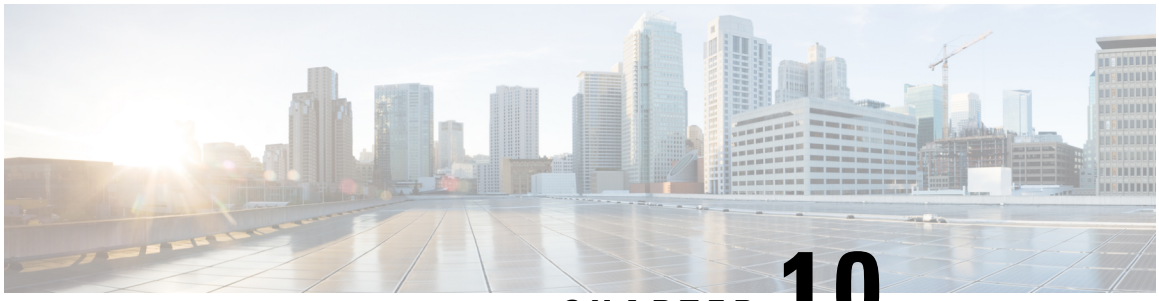
- Peer templates
- Route redistribution
- Route maps

Additional References

For additional information related to implementing BGP, see the following sections:

MIBs

MIBs	MIBs Link
BGP4-MIB CISCO-BGP4-MIB	To locate and download MIBs, go to the following URL: http://tools.cisco.com/ITDIT/MIBS/servlet/index



CHAPTER 10

Configuring Advanced BGP

This chapter describes how to configure advanced features of the Border Gateway Protocol (BGP) on Cisco NX-OS switches.

This chapter includes the following sections:

- Information About Advanced BGP, on page 217
- Prerequisites for Advanced BGP, on page 225
- Guidelines and Limitations for Advanced BGP, on page 226
- Default Settings for BGP, on page 227
- Configuring Advanced BGP, on page 227
- Configuring an Autonomous System Path Containing Your Own Autonomous System, on page 247
- Verifying the Advanced BGP Configuration, on page 260
- Displaying BGP Statistics, on page 261
- Related Topics, on page 262
- Additional References, on page 262

Information About Advanced BGP

BGP is an interdomain routing protocol that provides loop-free routing between organizations or autonomous systems. Cisco NX-OS supports BGP version 4. BGP version 4 includes multiprotocol extensions that allow BGP to carry routing information for IP routes and multiple Layer 3 protocol address families. BGP uses TCP as a reliable transport protocol to create TCP sessions with other BGP-enabled switches called BGP peers. When connecting to an external organization, the router creates external BGP (eBGP) peering sessions. BGP peers within the same organization exchange routing information through internal BGP (iBGP) peering sessions.

Peer Templates

BGP peer templates allow you to create blocks of common configurations that you can reuse across similar BGP peers. Each block allows you to define a set of attributes that a peer then inherits. You can choose to override some of the inherited attributes as well, making it a very flexible scheme for simplifying the repetitive nature of BGP configurations.

Cisco NX-OS implements three types of peer templates:

- The **peer-session** template defines BGP peer session attributes, such as the transport details, remote autonomous system number of the peer, and session timers. A peer-session template can also inherit attributes from another peer-session template (with locally defined attributes that override the attributes from an inherited peer-session).
- A **peer-policy** template defines the address-family dependent policy aspects for a peer including the inbound and outbound policy, filter-lists, and prefix-lists. A peer-policy template can inherit from a set of peer-policy templates. Cisco NX-OS evaluates these peer-policy templates in the order specified by the preference value in the inherit configuration. The lowest number is preferred over higher numbers.
- The **peer** template can inherit the peer-session and peer-policy templates to allow for simplified peer definitions. It is not mandatory to use a peer template but it can simplify the BGP configuration by providing reusable blocks of configuration.

Authentication

You can configure authentication for a BGP neighbor session. This authentication method adds an MD5 authentication digest to each TCP segment sent to the neighbor to protect BGP against unauthorized messages and TCP security attacks.

Note The MD5 password must be identical between BGP peers.

Route Policies and Resetting BGP Sessions

You can associate a route policy to a BGP peer. Route policies use route maps to control or modify the routes that BGP recognizes. You can configure a route policy for inbound or outbound route updates. The route policies can match on different criteria, such as a prefix or AS_path attribute, and selectively accept or deny the routes. Route policies can also modify the path attributes.

When you change a route policy applied to a BGP peer, you must reset the BGP sessions for that peer. Cisco NX-OS supports the following three mechanisms to reset BGP peering sessions:

- Hard reset—A hard reset tears down the specified peering sessions, including the TCP connection, and deletes routes coming from the specified peer. This option interrupts packet flow through the BGP network. Hard reset is disabled by default.
- Soft reconfiguration inbound—A soft reconfiguration inbound triggers routing updates for the specified peer without resetting the session. You can use this option if you change an inbound route policy. Soft reconfiguration inbound saves a copy of all routes received from the peer before processing the routes through the inbound route policy. If you change the inbound route policy, Cisco NX-OS passes these stored routes through the modified inbound route policy to update the route table without tearing down existing peering sessions. Soft reconfiguration inbound can use significant memory resources to store the unfiltered BGP routes. Soft reconfiguration inbound is disabled by default.
- Route Refresh—A route refresh updates the inbound routing tables dynamically by sending route refresh requests to supporting peers when you change an inbound route policy. The remote BGP peer responds with a new copy of its routes that the local BGP speaker processes with the modified route policy. Cisco NX-OS automatically sends an outbound route refresh of prefixes to the peer.

- BGP peers advertise the route refresh capability as part of the BGP capability negotiation when establishing the BGP peer session. Route refresh is the preferred option and enabled by default.

Note BGP also uses route maps for route redistribution, route aggregation, route dampening, and other features. SeeConfiguring Route Policy Manager, for more information on route maps.

eBGP

External BGP (eBGP) allows you to connect BGP peers from different autonomous systems to exchange routing updates. Connecting to external networks enables traffic from your network to be forwarded to other networks and across the Internet.

You should use loopback interfaces for establishing eBGP peering sessions because loopback interfaces are less susceptible to interface flapping. An interface flap occurs when the interface is administratively brought up or down because of a failure or maintenance issue. See the Configuring eBGP section for information on multihop, fast external failovers, and limiting the size of the AS-path attribute.

eBGP Next-Hop Unchanged

In an external BGP (eBGP) session, by default, the device changes the next-hop attribute of a BGP route (to its own address) when the device sends out a route. If the eBGP Next-Hop Unchanged feature is configured, BGP sends routes to an eBGP multihop peer without modifying the next-hop attribute. The next-hop attribute is unchanged. The BGP Next-hop Unchanged feature provides flexibility when designing and migrating networks. It can be used only between eBGP peers configured as multihop.

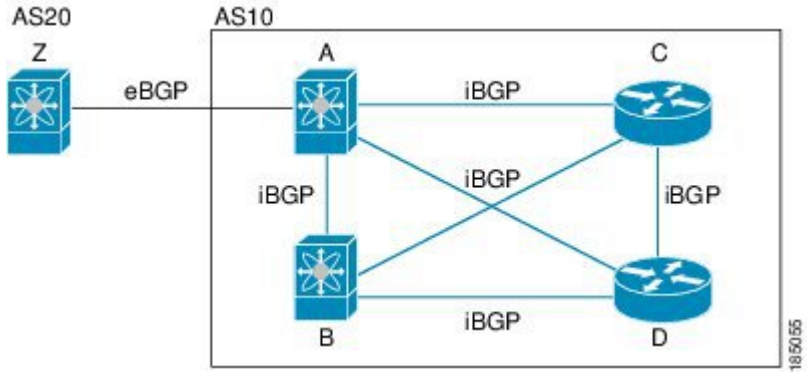
For example, consider a network with eBGP connection between Devices A, B, and C. Suppose Device A announces 100 prefixes to Device B. Device B is configured with an outbound route map to Device C and the match ip prefix list and set ip next-hop unchanged are configured on the route map. Device B propagates the unchanged next-hop address only for the routes that match the prefix list. For the other prefixes, it puts itself as the next-hop address.

iBGP

nternal BGP (iBGP) allows you to connect BGP peers within the same autonomous system. You can use iBGP for multihomed BGP networks (networks that have more than one connection to the same external autonomous system).

The following figure shows an iBGP network within a larger BGP network.

Figure 26: iBGP Network



iBGP networks are fully meshed. Each iBGP peer has a direct connection to all other iBGP peers to prevent network loops.

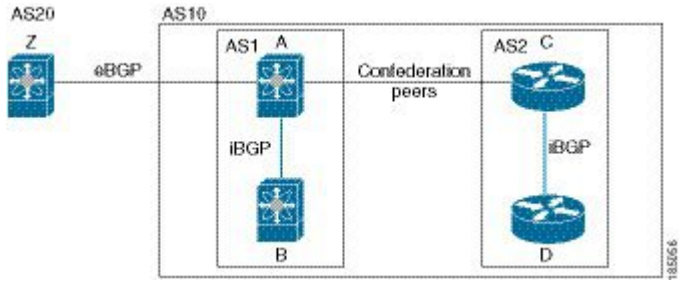
Note You should configure a separate interior gateway protocol in the iBGP network.

AS Confederations

A fully meshed iBGP network becomes complex as the number of iBGP peers grows. You can reduce the iBGP mesh by dividing the autonomous system into multiple subautonomous systems and grouping them into a single confederation. A confederation is a group of iBGP peers that use the same autonomous system number to communicate to external networks. Each subautonomous system is fully meshed within itself and has a few connections to other subautonomous systems in the same confederation.

The following figure shows the BGP network from Figure below, split into two subautonomous systems and one confederation.

Figure 27: AS Confederation



In this example, AS10 is split into two subautonomous systems, AS1 and AS2. Each subautonomous system is fully meshed, but there is only one link between the subautonomous systems. By using AS confederations, you can reduce the number of links compared to the fully meshed autonomous system in Figure **AS Confederation**.

Route Reflector

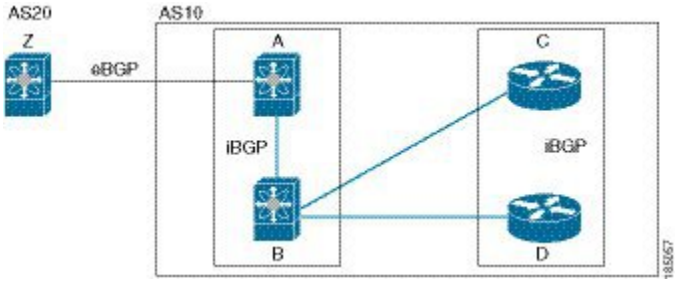
You can alternately reduce the iBGP mesh by using a route reflector configuration. Route reflectors pass learned routes to neighbors so that all iBGP peers do not need to be fully meshed.

Figure **iBGP Network** shows a simple iBGP configuration with four meshed iBGP speakers (router A, B, C, and D). Without route reflectors, when router A receives a route from an external neighbor, it advertises the route to all three iBGP neighbors.

When you configure an iBGP peer to be a route reflector, it becomes responsible for passing iBGP learned routes to a set of iBGP neighbors.

In the following figure, router B is the route reflector. When the route reflector receives routes advertised from router A, it advertises (reflects) the routes to routers C and D. Router A no longer has to advertise to both routers C and D.

Figure 28: Route Reflector



The route reflector and its client peers form a cluster. You do not have to configure all iBGP peers to act as client peers of the route reflector. You must configure any nonclient peer as fully meshed to guarantee that complete BGP updates reach all peers.

Capabilities Negotiation

A BGP speaker can learn about BGP extensions supported by a peer by using the capabilities negotiation feature. Capabilities negotiation allows BGP to use only the set of features supported by both BGP peers on a link.

If a BGP peer does not support capabilities negotiation, Cisco NX-OS will attempt a new session to the peer without capabilities negotiation if you have configured the address family as IPv4.

Route Dampening

Route dampening is a BGP feature that minimizes the propagation of flapping routes across an internetwork. A route flaps when it alternates between the available and unavailable states in rapid succession.

For example, consider a network with three BGP autonomous systems: AS1, AS2, and AS3. Suppose that a route in AS1 flaps (it becomes unavailable). Without route dampening, AS1 sends a withdraw message to AS2. AS2 propagates the withdrawal message to AS3. When the flapping route reappears, AS1 sends an advertisement message to AS2, which sends the advertisement to AS3. If the route repeatedly becomes unavailable, and then available, AS1 sends many withdrawal and advertisement messages that propagate through the other autonomous systems.

Route dampening can minimize flapping. Suppose that the route flaps. AS2 (in which route dampening is enabled) assigns the route a penalty of 1000. AS2 continues to advertise the status of the route to neighbors. Each time that the route flaps, AS2 adds to the penalty value. When the route flaps so often that the penalty exceeds a configurable suppression limit, AS2 stops advertising the route, regardless of how many times that it flaps. The route is now dampened.

The penalty placed on the route decays until the reuse limit is reached. At that time, AS2 advertises the route again. When the reuse limit is at 50 percent, AS2 removes the dampening information for the route.

Note The router does not apply a penalty to a resetting BGP peer when route dampening is enabled, even though the peer reset withdraws the route.

Load Sharing and Multipath

BGP can install multiple equal-cost eBGP or iBGP paths into the routing table to reach the same destination prefix. Traffic to the destination prefix is then shared across all the installed paths.

The BGP best-path algorithm considers the paths as equal-cost paths if the following attributes are identical:

- Weight
- Local preference
- AS_path
- Origin code
- Multi-exit discriminator (MED)
- IGP cost to the BGP next hop

BGP selects only one of these multiple paths as the best path and advertises the path to the BGP peers.

Note Paths received from different AS confederations are considered as equal-cost paths if the external AS_path values and the other attributes are identical.

Note When you configure a route reflector for iBGP multipath, and the route reflector advertises the selected best path to its peers, the next hop for the path is not modified.

Route Aggregation

You can configure aggregate addresses. Route aggregation simplifies route tables by replacing a number of more specific addresses with an address that represents all the specific addresses. For example, you can replace these three more specific addresses, 10.1.1.0/24, 10.1.2.0/24, and 10.1.3.0/24 with one aggregate address, 10.1.0.0/16.

Aggregate prefixes are present in the BGP route table so that fewer routes are advertised.

Note Cisco NX-OS does not support automatic route aggregation.

Route aggregation can lead to forwarding loops. To avoid this problem, when BGP generates an advertisement for an aggregate address, it automatically installs a summary discard route for that aggregate address in the local routing table. BGP sets the administrative distance of the summary discard to 220 and sets the route type to discard. BGP does not use discard routes for next-hop resolution.

BGP Conditional Advertisement

BGP conditional advertisement allows you to configure BGP to advertise or withdraw a route based on whether or not a prefix exists in the BGP table. This feature is useful, for example, in multihomed networks, in which you want BGP to advertise some prefixes to one of the providers only if information from the other provider is not present.

Consider an example network with three BGP autonomous systems: AS1, AS2, and AS3, where AS1 and AS3 connect to the Internet and to AS2. Without conditional advertisement, AS2 propagates all routes to both AS1 and AS3. With conditional advertisement, you can configure AS2 to advertise certain routes to AS3 only if routes from AS1 do not exist (if for example, the link to AS1 fails).

BGP conditional advertisement adds an exist or not-exist test to each route that matches the configured route map. See theConfiguring BGP Conditional Advertisement section for more information.

BGP Next-Hop Address Tracking

BGP monitors the next-hop address of installed routes to verify next-hop reachability and to select, install, and validate the BGP best path. BGP next-hop address tracking speeds up this next-hop reachability test by triggering the verification process when routes change in the RIB that may affect BGP next-hop reachability.

BGP receives notifications from the RIB when next-hop information changes (event-driven notifications). BGP is notified when any of the following events occurs:

- Next hop becomes unreachable.
- Next hop becomes reachable.
- Fully recursed IGP metric to the next hop changes.
- First hop IP address or first hop interface changes.
- Next hop becomes connected.
- Next hop becomes unconnected.
- Next hop becomes a local address.
- Next hop becomes a nonlocal address.

Note Reachability and recursed metric events trigger a best-path recalculation.

Event notifications from the RIB are classified as critical and noncritical. Notifications for critical and noncritical events are sent in separate batches. However, a noncritical event is sent with the critical events if the noncritical event is pending and there is a request to read the critical events.

- Critical events are related to the reachability (reachable and unreachable), connectivity (connected and unconnected), and locality (local and nonlocal) of the next hops. Notifications for these events are not delayed.
- Noncritical events include only the IGP metric changes.

See the Configuring BGP Next-Hop Address Tracking section for more information.

Site of Origin

The site of origin prevents routing loops when you have a multihomed VPN site. Routes learned from the same site are tagged with the same site-of-origin value that is configured at the PE on all the PE-CE links to the same site. Routes with a particular site-of-origin value are never readvertised back to a CE with the same site-of-origin value configured at the PE-CE link. This process prevents a CE router from relearning routes that originated from the same site. BGP and EIGRP use site of origin to prevent loops.

You can override the autonomous system number (ASN) of a site with the ASN of the provider. This feature is often used with the site of origin to identify the site where a route originated and prevent routing loops between routers within a VPN.

Route Redistribution

You can configure BGP to redistribute static routes or routes from other protocols. You configure a route policy with the redistribution to control which routes are passed into BGP. A route policy allows you to filter routes based on attributes such as the destination, origination protocol, route type, route tag, and so on. See Configuring Route Policy Manager, for more information.

BFD

This feature supports bidirectional forwarding detection (BFD). BFD is a detection protocol designed to provide fast forwarding-path failure detection times. BFD provides subsecond failure detection between two adjacent devices and can be less CPU-intensive than protocol hello messages because some of the BFD load can be distributed onto the data plane on supported modules.

BFD for BGP is supported on eBGP single-hop peers and iBGP single-hop peers. For iBGP single-hop peers using BFD, you must configure the update-source option in neighbor configuration mode. BFD is not supported on other iBGP peers or for multihop eBGP peers.

BFD is supported for the following types of interfaces:

- Layer 3 physical and subinterface
- Layer 3 port channel and subinterface
- Switch virtual interface (SVI)

BFD for BGP does not support authentication or per-link BFD sessions on a port channel.

See Configuring Bidirectional Forwarding Detection for more information.

Tuning BGP

You can modify the default behavior of BGP through BGP timers and by adjusting the best-path algorithm.

BGP Timers

BGP uses different types of timers for neighbor session and global protocol events. Each established session has a minimum of two timers for sending periodic keepalive messages and for timing out sessions when peer keepalives do not arrive within the expected time. In addition, there are other timers for handling specific features. Typically, you configure these timers in seconds. The timers include a random adjustment so that the same timers on different BGP peers trigger at different times.

Tuning the Best-Path Algorithm

You can modify the default behavior of the best-path algorithm through optional configuration parameters, including changing how the algorithm handles the multi-exit discriminator attribute and the router ID.

Multiprotocol BGP

BGP on Cisco NX-OS supports multiple address families. Multiprotocol BGP (MP-BGP) carries different sets of routes depending on the address family. For example, BGP can carry one set of routes for IPv4 unicast routing and one set of routes for IPv4 multicast routing. You can use MP-BGP for reverse-path forwarding (RPF) checks in IP multicast networks.

Note Because Multicast BGP does not propagate multicast state information, you need a multicast protocol, such as Protocol Independent Multicast (PIM).

Use the router address-family and neighbor address-family configuration modes to support multiprotocol BGP configurations. MP-BGP maintains separate RIBs for each configured address family, such as a unicast RIB and a multicast RIB for BGP.

A multiprotocol BGP network is backward compatible but BGP peers that do not support multiprotocol extensions cannot forward routing information, such as address family identifier information, that the multiprotocol extensions carry.

RFC 5549

BGP supports RFC 5549, which allows an IPv4 prefix to be carried over an IPv6 next hop. Because BGP is running on every hop, and all routers are capable of forwarding IPv4 and IPv6 traffic, there is no need to support IPv6 tunnels between any routers. BGP installs IPv4 over an IPv6 route to the Unicast Route Information Base (URIB)

Prerequisites for Advanced BGP

BGP has the following prerequisites:

- You must enable the BGP feature (see the Enabling the BGP Feature section).
- You should have a valid router ID configured on the system.

- You must have an AS number, either assigned by a Regional Internet Registry (RIR) or locally administered.
- You must have reachability (such as an interior gateway protocol (IGP), a static route, or a direct connection) to the peer that you are trying to make a neighbor relationship with.
- You must explicitly configure an address family under a neighbor for the BGP session establishment.

Guidelines and Limitations for Advanced BGP

BGP has the following configuration guidelines and limitations:

- Prefix peering operates only in passive TCP mode. It accepts incoming connections from remote peers if the peer address falls within the prefix.
- The dynamic AS number prefix peer configuration overrides the individual AS number configuration inherited from a BGP template.
- If you configure a dynamic AS number for prefix peers in an AS confederation, BGP establishes sessions with only the AS numbers in the local confederation.
- BGP sessions created through a dynamic AS number prefix peer ignore any configured eBGP multihop time-to-live (TTL) value or a disabled check for directly connected peers.
- Configure a router ID for BGP to avoid automatic router ID changes and session flaps.
- Use the maximum-prefix configuration option per peer to restrict the number of routes received and system resources used.
- Configure the update-source to establish a session with eBGP multihop sessions.
- Specify a BGP route map if you configure redistribution.
- Configure the BGP router ID within a VRF.
- If you decrease the keepalive and hold timer values, the network might experience session flaps.
- The following guidelines and limitations apply to the remove-private-as command.
  - If the local AS number of the device is a private AS number, you cannot use the remove-private-as configuration command for any other neighbor on the same device. As a workaround, you can use the local-as command on each neighbor with a public local AS number.
  - If the real AS number of the device is a private AS number and the remove-private-as all command is configured for a neighbor with a public local-as number, use local-as number [no-prepend [replace-as]] command to ensure that the real private AS number is not appended to the AS path.
  - If the real AS number of the device is a public AS number and the remove-private-as all command is configured for a neighbor, you cannot configure a private local-as number for the same neighbor. As a workaround, you must remove the existing configuration to proceed further.
  - The remove-private-as all command removes private AS numbers from the AS path even if the path contains both public and private AS numbers.
  - The remove-private-as command removes private AS numbers even if the AS path contains only private AS numbers. There is no likelihood of a 0-length AS path because this command can be

applied to eBGP peers only, in which case the AS number of the local device is appended to the AS path.

- The remove-private-as command removes private AS numbers even if the private AS numbers appear before the confederation segments in the AS path.
- When you remove private AS numbers from the AS path, the path length of the prefixes that are sent out will decrease. Because the AS path length is a key element of BGP best-path selection, it might be necessary to retain the path length. The replace-as keyword ensures that the path length is retained by replacing all removed AS numbers with the local router's AS number.

Default Settings for BGP

Table below lists the default settings for BGP parameters.

Table 17: Default BGP Parameters

Parameters	Default
BGP feature	disabled
keep alive interval	60 seconds
hold timer	180 seconds

Configuring Advanced BGP

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

Enabling IP Forward on an Interface

To use RFC 5549, you must configure at least one IPv4 address. If you do not want to configure an IPv4 address, you must enable the ip forward feature to use RFC 5549.

Procedure

	Command or Action	Purpose

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

	Command or Action	Purpose

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

Step 3	**ip forward**
	Example:
	`switch(config-if)# ip forward`
	`switch(config-if)#`
		Allows IPv4 traffic on the interface even when there is no IP address configuration on that interface.

Example

This example shows how to enable the ip forward feature on an interface:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ip forward
```

Configuring BGP Session Templates

You can use BGP session templates to simplify BGP configuration for multiple BGP peers with similar configuration needs. BGP templates allow you to reuse common configuration blocks. You configure BGP templates first, and then apply these templates to BGP peers.

With BGP session templates, you can configure session attributes such as inheritance, passwords, timers, and security.

A peer-session template can inherit from one other peer-session template. You can configure the second template to inherit from a third template. The first template also inherits this third template. This indirect inheritance can continue for up to seven peer-session templates.

Any attributes configured for the neighbor take priority over any attributes inherited by that neighbor from a BGP template.

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

When editing a template, you can use the **no** form of a command at either the peer or template level to explicitly override a setting in a template. You must use the **default** form of the command to reset that attribute to the default state.

Procedure

	Command or Action	Purpose

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

	Command or Action	Purpose

Step 2	**router bgp** *autonomous-system-number*
	Example:
	`switch(config)# router bgp 65536`
	`switch(config-router)#`
		Enables BGP and assigns the autonomous system number to the local BGP speaker.

Step 3	**template peer-session** *template-name*
	Example:
	`switch(config-router)# template peer-session BaseSession`
	`switch(config-router-stmp)#`
		Enters peer-session template configuration mode.

Step 4	(Optional) **password** *number password* **Example:** `switch(config-router-stmp)# password 0 test`	Adds the cleartext password *test* to the neighbor. The password is stored and displayed in type 3 encrypted form (3DES).
Step 5	(Optional) **timers** *keepalive hold* **Example:** `switch(config-router-stmp)# timers 30 90`	Adds the BGP keepalive and holdtimer values to the peer-session template. The default keepalive interval is 60. The default hold time is 180.

Step 6	**exit**
	Example:
	`switch(config-router-stmp)# exit`
	`switch(config-router)#`
		Exits peer-session template configuration mode.

Step 7	**neighbor** *ip-address* **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 192.168.1.2 remote-as 65536`
	`switch(config-router-neighbor)#`
		Places the router in the neighbor configuration mode for BGP routing and configures the neighbor IP address.

Step 8	**inherit peer-session** *template-name*
	Example:
	`switch(config-router-neighbor)# inherit peer-session BaseSession`
	`switch(config-router-neighbor)`
		Applies a peer-session template to the peer.

Step 9	(Optional) **description** *text*
	Example:
	`switch(config-router-neighbor)# description Peer Router A`
	`switch(config-router-neighbor)`
		Adds a description for the neighbor.

Step 10	(Optional) **show bgp peer-session** *template-name* **Example:**	Displays the peer-policy template.

	Command or Action	Purpose
	`switch(config-router-neighbor)# show bgp peer-session BaseSession`	
Step 11	(Optional) copy running-config startup-config **Example:** `switch(config-router-neighbor)# copy running-config startup-config`	Saves this configuration change.

Example

Use the **show bgp neighbor** command to see the template applied. See the Cisco Nexus 3000 Series Command Reference for details on all commands available in the template.

This example shows how to configure a BGP peer-session template and apply it to a BGP peer:

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# template peer-session BaseSession
switch(config-router-stmp)# timers 30 90
switch(config-router-stmp)# exit
switch(config-router)# neighbor 192.168.1.2 remote-as 65536
switch(config-router-neighbor)# inherit peer-session BaseSession
switch(config-router-neighbor)# description Peer Router A
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor)# copy running-config startup-config
```

Configuring BGP Peer-Policy Templates

You can configure a peer-policy template to define attributes for a particular address family. You assign a preference to each peer-policy template and these templates are inherited in the order specified, for up to five peer-policy templates in a neighbor address family.

Cisco NX-OS evaluates multiple peer policies for an address family using the preference value. The lowest preference value is evaluated first. Any attributes configured for the neighbor take priority over any attributes inherited by that neighbor from a BGP template.

Peer-policy templates can configure address family-specific attributes such as AS-path filter lists, prefix lists, route reflection, and soft reconfiguration.

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Note When editing a template, you can use the **no** form of a command at either the peer or template level to explicitly override a setting in a template. You must use the default form of the command to reset that attribute to the default state.

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *autonomous-system-number*
	Example:
	`switch(config)# router bgp 65536`
	`switch(config-router)#`
		Enables BGP and assigns the autonomous system number to the local BGP speaker.

Step 3	**template peer-policy** *template-name*
	Example:
	`switch(config-router)# template peer-policy BasePolicy`
	`switch(config-router-ptmp)#`
		Creates a peer-policy template.

Step 4	(Optional) **advertise-active-only** **Example:** `switch(config-router-ptmp)# advertise-active-only`	Advertises only active routes to the peer.
Step 5	(Optional) **maximum-prefix** *number* **Example:** `switch(config-router-ptmp)# maximum-prefix 20`	Sets the maximum number of prefixes allowed from this peer.

Step 6	**exit**
	Example:
	`switch(config-router-ptmp)# exit`
	`switch(config-router)#`
		Exits peer-policy template configuration mode.

Step 7	**neighbor** *ip-address* **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 192.168.1.2 remote-as 65536`
	`switch(config-router-neighbor)#`
		Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.

Step 8	**address-family** { **ipv4** | **ipv6**} { **multicast** | **unicast** }
	Example:
	`switch(config-router-neighbor)# address-family ipv4 unicast`
	`switch(config-router-neighbor-af)#`
		Enters global address family configuration mode for the specified address family.

	Command or Action	Purpose
Step 9	**inherit peer-policy** *template-name preference* **Example:** `switch(config-router-neighbor-af)# inherit peer-policy BasePolicy 1`	Applies a peer-policy template to the peer address family configuration and assigns the preference value for this peer policy.
Step 10	(Optional) **show bgp peer-policy** *template-name* **Example:** `switch(config-router-neighbor-af)# show bgp peer-policy BasePolicy`	Displays the peer-policy template.
Step 11	(Optional) copy running-config startup-config **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	Saves this configuration change.

Example

Use the **show bgp neighbor** command to see the template applied. See the Cisco Nexus 3000 Series Command Reference for details on all commands available in the template.

This example shows how to configure a BGP peer-session template and apply it to a BGP peer:

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# template peer-session BasePolicy
switch(config-router-ptmp)# maximum-prefix 20
switch(config-router-ptmp)# exit
switch(config-router)# neighbor 192.168.1.1 remote-as 65536
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# inherit peer-policy BasePolicy
switch(config-router-neighbor-af)# copy running-config startup-config
```

Configuring BGP Peer Templates

You can configure BGP peer templates to combine session and policy attributes in one reusable configuration block. Peer templates can also inherit peer-session or peer-policy templates. Any attributes configured for the neighbor take priority over any attributes inherited by that neighbor from a BGP template. You configure only one peer template for a neighbor, but that peer template can inherit peer-session and peer-policy templates.

Peer templates support session and address family attributes, such as eBGP multihop time-to-live, maximum prefix, next-hop self, and timers.

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Note When editing a template, you can use the **no** form of a command at either the peer or template level to explicitly override a setting in a template. You must use the default form of the command to reset that attribute to the default state.

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *autonomous-system-number* **Example:** `switch(config)# router bgp 65536`	Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

Step 3	**template peer** *template-name*
	Example:
	`switch(config-router)# template peer BasePeer`
	`switch(config-router-neighbor)#`
		Enters peer template configuration mode.

Step 4	(Optional) **inherit peer-session** *template-name* **Example:** `switch(config-router-neighbor)# inherit peer-session BaseSession`	Inherits a peer-session template in the peer template.

Step 5	(Optional) **address-family** { **ipv4** | **ipv6**} { **multicast** | **unicast** }
	Example:
	`switch(config-router-neighbor)# address-family ipv4 unicast`
	`switch(config-router-neighbor-af)#`
		Configures the global address family configuration mode for the specified address family.

Step 6	(Optional) **inherit peer** *template-name* **Example:** `switch(config-router-neighbor-af)# inherit peer BasePolicy`	Applies a peer template to the neighbor address family configuration.

Step 7	**exit**
	Example:
	`switch(config-router-neighbor-af)# exit`
	`switch(config-router-neighbor)#`
		Exits BGP neighbor address family configuration mode.

Step 8	(Optional) **timers** *keepalive hold* **Example:**	Adds the BGP timer values to the peer.

	Command or Action	Purpose
	`switch(config-router-neighbor)# timers 45 100`	These values override the timer values in the peer-session template, BaseSession.

Step 9	**exit**
	Example:
	`switch(config-router-neighbor)# exit`
	`switch(config-router)#`
		Exits BGP peer template configuration mode.

Step 10	**neighbor** *ip-address* **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 192.168.1.2 remote-as 65536`
	`switch(config-router-neighbor)#`
		Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.

Step 11	**inherit peer** *template-name* **Example:** `switch(config-router-neighbor)# inherit peer BasePeer`	Inherits the peer template.
Step 12	(Optional) **timers** *keepalive hold* **Example:** `switch(config-router-neighbor)# timers 60 120`	Adds the BGP timer values to this neighbor. These values override the timer values in the peer template and the peer-session template.
Step 13	(Optional) **show bgp peer-template** *template-name* **Example:** `switch(config-router-neighbor-af)# show bgp peer-template BasePeer`	Displays the peer template.
Step 14	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	Saves this configuration change.

Example

Use the **show bgp neighbor** command to see the template applied. See the Cisco Nexus 3600 Series Command Reference for details on all commands available in the template.

This example shows how to configure a BGP peer template and apply it to a BGP peer:

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# template peer BasePeer
switch(config-router-neighbor)# inherit peer-session BaseSession
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# inherit peer-policy BasePolicy 1
switch(config-router-neighbor-af)# exit
```

```
switch(config-router-neighbor)# exit
switch(config-router)# neighbor 192.168.1.2 remote-as 65536
switch(config-router-neighbor)# inherit peer BasePeer
switch(config-router-neighbor)# copy running-config startup-config
```

Configuring Prefix Peering

BGP supports the definition of a set of peers using a prefix for both IPv4 and IPv6. This feature allows you to not have to add each neighbor to the configuration.

When defining a prefix peering, you must specify the remote AS number with the prefix. BGP accepts any peer that connects from that prefix and autonomous system if the prefix peering does not exceed the configured maximum peers allowed.

When a BGP peer that is part of a prefix peering disconnects, Cisco NX-OS holds its peer structures for a defined prefix peer timeout value. An established peer can reset and reconnect without danger of being blocked because other peers have consumed all slots for that prefix peering.

To configure the BGP prefix peering timeout value, use the following command in router configuration mode:

Command	Purpose
timers prefix-peer-timeout *value* **Example** : `switch(config-router)# timers prefix-peer-timeout 120`	Configures the timeout value for prefix peering. The range is from 0 to 1200 seconds. The default value is 30. **Note** For prefix peers, set the prefix peer timeout to be greater than the configured graceful restart timer. If the prefix peer timeout is greater than the graceful restart timer, a peer's route is retained during its restart. If the prefix peer timeout is less than the graceful restart timer, the peer's route is purged by the prefix peer timeout, which may occur before the restart is complete.
timers prefix-peer-wait interval **Example** : `switch(config-router)# timers prefix-peer-wait 50`	Configures the BGP prefix peering wait timer on a per-VRF basis or on the default VRF. You can use the timers prefix-peer-wait command to disable the peer prefix wait time so that there is no delay before BGP prefixes are inserted into the routing information base (RIB). The range of the interval is from 0 to 1200 seconds. The default value is 90 seconds. **Note** The timer is only applicable for BGP dynamic neighbors. It is only set when BGP is restarted or is coming up for the first time for dynamic BGP neighbors.

To configure the maximum number of peers, use the following command in neighbor configuration mode:

Command	Purpose
maximum-peers *value* **Example** : `switch(config-router-neighbor)# maximum-peers 120`	Configures the maximum number of peers for this prefix peering. The range is from 1 to 1000.

This example shows how to configure a prefix peering that accepts up to 10 peers:

```
switch(config)# router bgp 65536
switch(config-router)# timers prefix-peer-timeout 120
switch(config-router)# neighbor 10.100.200.0/24 remote-as 65536
switch(config-router-neighbor)# maximum-peers 10
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)#
```

This example shows how to disable the peer prefix wait time:

```
switch(config)# router bgp 100
switch(config-router)# timers prefix-peer-wait 50
switch(config-router)#
```

Use the **show ip bgp neighbors** command to show the details of the configuration for that prefix peering with a list of the currently accepted instances and the counts of active, maximum concurrent, and total accepted peers.

Configuring BGP Authentication

You can configure BGP to authenticate route updates from peers using MD5 digests.

To configure BGP to use MD5 authentication, use the following command in neighbor configuration mode:

Command	Purpose
password [**0** | **3** | **7**] *string* **Example** : `switch(config-router-neighbor)# password BGPpassword`	Configures an MD5 password for BGP neighbor sessions.

Resetting a BGP Session

If you modify a route policy for BGP, you must reset the associated BGP peer sessions. If the BGP peers do not support route refresh, you can configure a soft reconfiguration for inbound policy changes. Cisco NX-OS automatically attempts a soft reset for the session.

To configure soft reconfiguration inbound, use the following command in neighbor address-family configuration mode:

Command	Purpose
soft-reconfiguration inbound always **Example** : `switch(config-router-neighbor-af)# soft-reconfiguration inbound always`	Enables soft reconfiguration to store the inbound BGP route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.

To reset a BGP neighbor session, use the following command in any mode:

Command	Purpose
clear bgp ip { **unicast** | **multicast** } *ip-address* **soft** { **in** | **out** } **Example** : `switch# clear bgp ip unicast 192.0.2.1 soft in`	Resets the BGP session without tearing down the TCP session.

Modifying the Next-Hop Address

You can modify the next-hop address used in a route advertisement in the following ways:

- Disable the next-hop calculation and use the local BGP speaker address as the next-hop address.
- Set the next-hop address as a third-party address. Use this feature in situations where the original next-hop address is on the same subnet as the peer that the route is being sent to. Using this feature saves an extra hop during forwarding.

To modify the next-hop address, use the following parameters in commands address-family configuration mode:

Command	Purpose
next-hop-self **Example** : `switch(config-router-neighbor-af)# next-hop-self`	Uses the local BGP speaker address as the next-hop address in route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
next-hop-third-party **Example** : `switch(config-router-neighbor-af)# next-hop-third-party`	Sets the next-hop address as a third-party address. Use this command for single-hop EBGP peers that do not have **next-hop-self** configured.

Configuring BGP Next-Hop Address Tracking

BGP next-hop address tracking is enabled by default and cannot be disabled.

You can modify the delay interval between RIB checks to increase the performance of BGP next-hop tracking. You can configure the critical timer for routes that affect BGP next-hop reachability, and you can configure the noncritical timer for all other routes in the BGP table.

To modify the BGP next-hop address tracking, use the following commands address-family configuration mode:

Command	Purpose
nexthop trigger-delay {**critical** | **non-critical** } *milliseconds* **Example** : `switch(config-router-af)# nexthop trigger-delay critical 5000`	Specifies the next-hop address tracking delay timer for critical next-hop reachability routes and for noncritical routes. The range is from 1 to 4294967295 milliseconds. The critical timer default is 3000. The noncritical timer default is 10000.
nexthop route-map *name* **Example** : `switch(config-router-af)# nexthop route-map nextHopLimits`	Specifies a route map to match the BGP next-hop addresses to. The name can be any case-sensitive, alphanumeric string up to 63 characters.

Configuring Next-Hop Filtering

BGP next-hop filtering allows you to specify that when a next-hop address is checked with the RIB, the underlying route for that next-hop address is passed through the route map. If the route map rejects the route, the next-hop address is treated as unreachable.

BGP marks all next-hops that are rejected by the route policy as invalid and does not calculate the best path for the routes that use the invalid next-hop address.

To configure BGP next-hop filtering, use the following command in address-family configuration mode:

Command	Purpose
nexthop route-map *name* **Example** : `switch(config-router-af)# nexthop route-map nextHopLimits`	Specifies a route map to match the BGP next-hop route to. The name can be any case-sensitive, alphanumeric string up to 63 characters.

Controlling Reflected Routes Through Next-Hop-Self

NX-OS enables controlling the iBGP routes being sent to a specific peer through the **next-hop-self** [all] arguments. By using these arguments, you can selectively change the next-hop of routes even if the route is reflected.

Command	Purpose
next-hop-self [all] **Example**: `switch(config-router-af)# next-hop-self all`	Uses the local BGP speaker address as the next-hop address in route updates. The all keyword is optional. If you specify all, all routes are sent to the peer with next-hop-self. If you do not specify all, the next hops of reflected routes are not changed.

Shrinking Next-Hop Groups When A Session Goes Down

You can configure BGP to shrink ECMP groups in an accelerated way when a session goes down.

This feature applies to the following BGP path failure events:

- Any single or multiple Layer 3 link failures
- BFD failure detections for BGP neighbors
- Administrative shutdown of BGP neighbors (using the shutdown command)

The accelerated handling of Layer 3 link failures is enabled by default and does not require a configuration command to be enabled.

To configure the accelerated handling of the last two events, use the following command in the router configuration mode:

Command	Purpose
neighbor-down fib-accelerate **Example** : `switch(config-router)# neighbor-down fib-accelerate`	Withdraws the corresponding next hop from all next-hop groups (ECMP groups and single next-hop routes) whenever a BGP session goes down. **Note** This command applies to both IPv4 and IPv6 address-family routes and is supported only in a BGP-only environment where all non-direct routes are installed by BGP.

Disabling Capabilities Negotiation

You can disable capabilities negotiations to interoperate with older BGP peers that do not support capabilities negotiation.

To disable capabilities negotiation, use the following command in neighbor configuration mode:

Command	Purpose
dont-capability-negotiate **Example** : `switch(config-router-neighbor)# dont-capability-negotiate`	Disables capabilities negotiation. You must manually reset the BGP sessions after configuring this command.

Configuring BGP Additional Paths

GP supports sending and receiving multiple paths per prefix and advertising such paths.

Advertising the Capability of Sending and Receiving Additional Paths

You can configure BGP to advertise the capability of sending and receiving additional paths to and from the BGP peers. To do so, use the following commands in neighbor address-family configuration mode:

Procedure

	Command or Action	Purpose
Step 1	[**no**] **capability additional-paths send** [**disable**] **Example:** `switch(config-router-neighbor-af)# capability addtional-paths send`	Advertises the capability to send additional paths to the BGP peer. The **disable** option disables the advertising capability of sending additional paths. The **no** form of this command disables the capability of sending additional paths.
Step 2	[**no**] **capability additional-paths receive** [**disable**] **Example:** `switch(config-router-neighbor-af)# capability addtional-paths receive`	Advertises the capability to receive additional paths from the BGP peer. The **disable** option disables the advertising capability of receiving additional paths. The **no** form of this command disables the capability of receiving additional paths.
Step 3	**show bgp neighbor** **Example:** `switch(config-router-neighbor-af)# show bgp neighbor`	Displays whether the local peer has advertised the additional paths send or receive capability to the remote peer.

Example

This example shows how to configure BGP to advertise the capability to send and receive additional paths to and from the BGP peer:

```
switch# configure terminal
switch(config)# router bgp 100
switch(config-router)# neighbor 10.131.31.2 remote-as 100
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# capability additional-paths send
switch(config-router-neighbor-af)# capability additional-paths receive
```

Configuring the Sending and Receiving of Additional Paths

You can configure the capability of sending and receiving additional paths to and from the BGP peers. To do so, use the following commands in address-family configuration mode:

Procedure

	Command or Action	Purpose
Step 1	[**no**] **additional-paths send** **Example:** `switch(config-router-af)# additional-paths send`	Enables the send capability of additional paths for all of the neighbors under this address family for which the capability has not been disabled. The **no** form of this command disables the send capability.

	Command or Action	Purpose
Step 2	[**no**] **additional-paths receive** **Example:** `switch(config-router-af)# additional-paths receive`	Enables the receive capability of additional paths for all of the neighbors under this address family for which the capability has not been disabled. The **no** form of this command disables the receive capability.
Step 3	**show bgp neighbor** **Example:** `switch(config-router-af)# show bgp neighbor`	Displays whether the local peer as advertised the additional paths send or receive capability to the remote peer.

Example

This example shows how to enable the additional paths send and receive capability for all neighbors under the specified address family for which this capability has not been disabled:

```
switch# configure terminal
switch(config)# router bgp 100
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# additional-paths send
switch(config-router-af)# additional-paths receive
```

Configuring Advertised Paths

You can specify the paths that are advertised for BGP. To do so, use the following commands in route-map configuration mode:

Procedure

	Command or Action	Purpose
Step 1	[**no**] **set ip next-hop unchanged** **Example:** `switch(config-route-map)# set ip next-hop unchanged`	Specifies and unchanged next-hop IP address.
Step 2	[**no**] **set path-selection all** | advertise **Example:** `switch(config-route-map)# set path-selection all advertise`	Specifies that all paths be advertised for a given prefix. • all—Advertises all available valid paths. The **no** form of this command specifies that only the best path be advertised.
Step 3	**show bgp** {**ipv4** | **ipv6**} **unicast** [*ip-address* | *ipv6-prefix*] [**vrf** *vrf-name*] **Example:**	Displays the path ID for the additional paths of a prefix and advertisement information for these paths.

	Command or Action	Purpose
	`switch(config-route-map)# show bgp ipv4 unicast`	

Example

This example show how to specify that all paths be advertised for the prefix list p1:

```
switch# configure terminal
switch(config)# route-map PATH_SELECTION_RMAP
switch(config-route-map)# match ip address prefix-list p1
switch(config-route-map)# set path-selection all advertise
```

Configuring Additional Path Selection

You can configure the capability fo selecting additional paths for a prefix. To do so, use the following commands in address-family configuration mode:

Procedure

	Command or Action	Purpose

Step 1	[**no**] **additional-paths selection route-map** *map-name*
	Example:
	`switch(config-router-af)# additional paths`
	`selection route-map map1`
		Configures the capability of selecting additional paths for a prefix.
		The **no** form of this command disables the additional paths selection capability.

Step 2	**show bgp** {**ipv4** | **ipv6**} **unicast** [*ip-address* | *ipv6-prefix*] [**vrf** *vrf-name*] **Example:** `switch(config-route-af)# show bgp ipv4 unicast`	Displays the path ID for the additional paths of a prefix and advertisement information for these paths.

Example

This example shows how to configure additional paths selection under the specified address family:

```
switch# configure terminal
switch(config)# router bgp 100
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# additional-paths selection route-map PATH_SELECTION_RMAP
```

Configuring eBGP

This section includes the following topics:

Configuring eBGP Next-Hop Unchanged

You can configure eBGP to send routes to an eBGP multihop peer without changing the next-hop address. By default, the device changes the next-hop address of a BGP route to its own address when the device sends out a route.

	Command	Purpose
Step 1	**disable-connected-check** **Example** : `switch(config-router-neighbor)# disable-connected-check`	Disables checking whether or not a single-hop eBGP peer is directly connected. You must manually reset the BGP sessions after using this command.
Step 2	**configure terminal** Example: `switch# configure terminal`	Enters global configuration mode.
Step 3	**route-map** *route-map name* Example: `switch(config)# route-map route`	Enters route map configuration mode.
Step 4	**set ip next-hop unchanged** Example: `switch(config-route-map)# set ip next-hop unchanged`	Configures the device to send BGP updates to the specified eBGP peer without modifying the next-hop address.
Step 5	**exit** Example: `switch(config-route-map)# exit`	Exits route map configuration mode.

This example shows how to set eBGP next-hop unchanged to send routes without changing the next-hop address:

```
switch# configure terminal
switch(config)# route-map route
switch(config-route-map)# set ip next-hop unchanged
switch(config-route-map)# exit
switch(config)#
```

Disabling eBGP Single-Hop Checking

You can configure eBGP to disable checking whether a single-hop eBGP peer is directly connected to the local router. Use this option for configuring a single-hop loopback eBGP session between directly connected switches.

To disable checking whether or not a single-hop eBGP peer is directly connected, use the following command in neighbor configuration mode:

Command	Purpose
disable-connected-check **Example** : `switch(config-router-neighbor)# disable-connected-check`	Disables checking whether or not a single-hop eBGP peer is directly connected. You must manually reset the BGP sessions after using this command.

Configuring eBGP Multihop

You can configure the eBGP time-to-live (TTL) value to support eBGP multihop. In some situations, an eBGP peer is not directly connected to another eBGP peer and requires multiple hops to reach the remote eBGP peer. You can configure the eBGP TTL value for a neighbor session to allow these multihop sessions.

To configure eBGP multihop, use the following command in neighbor configuration mode:

Command	Purpose
ebgp-multihop *ttl-value* **Example** : `switch(config-router-neighbor)# ebgp-multihop 5`	Configures the eBGP TTL value for eBGP multihop. The range is from 2 to 255. You must manually reset the BGP sessions after using this command.

Configuring eBGP Routes in the Same Autonomous System

You can configure eBGP learned routes from a remote autonomous system (AS) to advertise to another eBGP peer in the same AS.

Note When route updates are sent between peers within the same AS number, they are dropped unless you enter the **allowas-in** command.

To disable AS peer checking, use the following command in neighbor configuration mode:

	Command	Purpose
Step 1	**router bgp** *autonomous-system-number* **Example** : `switch(config)# router bgp 64496`	Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 2	**neighbor ipv4 remote-as** *as-number* **Example** : `switch (config-router)# neighbor 209.165.201.1 remote-as 64497`	Configures the specified address type and AS number for a remote BGP peer. The ip-address format is x.x.x.x. The IPv6 address-format is A:B::C:D.

	Command	Purpose
Step 3	**address family ipv4 unicast** **Example** : `switch(config-router)# address family ip4 unicast`	Enters neighbor address family configuration mode for the unicast specified address family.
Step 4	**disable-peer-as-check** **Example** : `switch(config-router-neighbor-af)# disable-peer-as-check`	Disables AS checking so that routes are updated between peers in the same AS.
Step 5	**show bgp neighbor** **Example** : `switch(config-router-neighbor-af)# show bgp ipv4 unicast neighbors`	Displays information about BGP peers.

This example shows how to display BGP peer information:

```
switch(config)# show bgp neighbor 1.222.222.2
bgp neighbor is 1.222.222.2, remote as 2222, ebgp link, peer index 1
bgp version 4, remote router id1.100.1.2 ####output trunicated####
for address family:ipv4 unicast
bgp table version 54, neighbor version 54
3 accepted paths consume 108 bytes of memory
10 sent paths
peer asn check is disabled

#####output omitted#####
```

Disabling a Fast External Failover

Typically, when a BGP router loses connectivity to a directly connected eBGP peer, BGP triggers a fast external failover by resetting the eBGP session to the peer. You can disable this fast external failover to limit the instability caused by link flaps.

To disable a fast external failover, use the following command in router configuration mode:

Command	Purpose
no fast-external-fallover **Example** : `switch(config-router)# no fast-external-fallover`	Disables a fast external failover for eBGP peers. This command is enabled by default.

Limiting the AS-path Attribute

You can configure eBGP to discard routes that have a high number of AS numbers in the AS-path attribute.

To discard routes that have a high number of AS numbers in the AS-path attribute, use the following command in router configuration mode:

Command	Purpose
maxas-limit *number* **Example** : `switch(config-router)# maxas-limit 50`	Discards eBGP routes that have a number of AS-path segments that exceed the specified limit. The range is from 1 to 2000 .

Configuring Local AS Support

The local AS feature allows a router to appear to be a member of a second autonomous system (AS), in addition to its real AS. Local AS allows two ISPs to merge without modifying peering arrangements. Routers in the merged ISP become members of the new autonomous system but continue to use their old AS numbers for their customers.

Local AS can only be used for true eBGP peers. You cannot use this feature for two peers that are members of different confederation subautonomous systems.

To configure eBGP local AS support, use the following command in neighbor configuration mode:

Command	Purpose
local-as *number* [**no-prepend** [**replace-as** [**dual-as**]]] **Example** : `switch(config-router-neighbor)# local-as 1.1`	Configures eBGP to prepend the local AS *number* to the AS_PATH attribute. The **local-as** *number* can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format. The **no-prepend** keyword ensures that the **local-as** *number* is not prepended to any downstream BGP neighbors except for the partner who is peering with the **local-as** *number*. The **replace-as** keyword ensures that only the **local-as** *number* of the peering session is prepended to the AS_PATH attribute. The autonomous-system number from the local BGP routing process is not prepended. The **dual-as** keyword configures the eBGP neighbor to establish a peering session using the real autonomous-system number (from the local BGP routing process) or by using the autonomous-system number configured as the Local AS).

Configuring AS Confederations

To configure an AS confederation, you must specify a confederation identifier. The group of autonomous systems within the AS confederation looks like a single autonomous system with the confederation identifier as the autonomous system number.

To configure a BGP confederation identifier, use the following command in router configuration mode:

Command	Purpose
confederation identifier *as-number* **Example** : `switch(config-router)# confederation identifier 64512`	Configures a confederation identifier for an AS confederation. Each confederation has a different sub-AS number, usually a private one (from 64512 to 65534). This command triggers an automatic notification and session reset for the BGP neighbor sessions.

To configure the autonomous systems that belong to the AS confederation, use the following command in router configuration mode:

Command	Purpose
bgp confederation peers *as-number* [*as-number2...*] **Example** : `switch(config-router)# bgp confederation peers 5 33 44`	Specifies a list of autonomous systems that belong to the confederation. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

Configuring an Autonomous System Path Containing Your Own Autonomous System

Enable the feature for BGP to accept the autonomous system (AS) path that contains your own AS.

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**router bgp** *as-number*	Enters BGP mode and assigns the autonomous system number to the local BGP speaker. The *as-number* value range is from 1 to 65535.
Step 3	**neighbor** *ip-address* **remote-as** *as-number*	Enters neighbor configuration mode for BGP routing and configures the neighbor IP address.
Step 4	**address-family** { **ipv4** | **ipv6**} { **multicast** | **unicast** }	Enters router address family configuration for the specified address family.
Step 5	[**no** | **default**] **allowas-in** [*allowas-in-cnt*]	Enables the allowas-in feature for BGP and configures the number of occurrences of the AS number.

	Command or Action	Purpose
		For **allowas-in-cnt**, enter an integer between 1 and 10. By default, the number of occurrences of the AS number is set to 3.
Step 6	**end**	Exits router address family configuration mode.
Step 7	(Optional) **show running-config bgp**	Displays the BGP configuration.
Step 8	**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 allowas-in feature for BGP and configure it for a unicast IPv4 address family:

```
switch# configure terminal
switch(config)# router bgp 77
switch(config-router)# neighbor 6.20.1.1 remote-as 66
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# allowas-in 5
switch(config-router-neighbor-af)# end
```

Configuring Route Reflector

You can configure iBGP peers as route reflector clients to the local BGP speaker, which acts as the route reflector. Together, a route reflector and its clients form a cluster. A cluster of clients usually has a single route reflector. In such instances, the cluster is identified by the router ID of the route reflector. To increase redundancy and avoid a single point of failure in the network, you can configure a cluster with more than one route reflector. You must configure all route reflectors in the cluster with the same 4-byte cluster ID so that a route reflector can recognize updates from route reflectors in the same cluster.

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *as-number*
	Example:
	`switch(config)# router bgp 65536`
	`switch(config-router)#`
		Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

	Command or Action	Purpose
Step 3	**cluster-id** *cluster-id* **Example:** `switch(config-router)# cluster-id 192.0.2.1`	Configures the local router as one of the route reflectors that serve the cluster. You specify a cluster ID to identify the cluster. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.

Step 4	**address-family** {**ipv4** | **ipv6**} { **unicast** | **multicast**}
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters router address family configuration mode for the specified address family.

Step 5	(Optional) **client-to-client reflection** **Example:** `switch(config-router-af)# client-to-client reflection`	Configures client-to-client route reflection. This feature is enabled by default. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.

Step 6	**exit**
	Example:
	`switch(config-router-neighbor)# exit`
	`switch(config-router)#`
		Exits router address configuration mode.

Step 7	**neighbor** *ip-address* **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 192.0.2.10 remote-as 65536`
	`switch(config-router-neighbor)#`
		Configures the IP address and AS number for a remote BGP peer.

Step 8	**address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }
	Example:
	`switch(config-router-neighbor)# address-family ipv4 unicast`
	`switch(config-router-neighbor-af)#`
		Enters neighbor address family configuration mode for the specified address family.

Step 9	**route-reflector-client** **Example:** `switch(config-router-neighbor-af)# route-reflector-client`	Configures the switch as a BGP route reflector and configures the neighbor as its client. This command triggers an automatic notification and session reset for the BGP neighbor sessions.
Step 10	(Optional) **show bgp ip** { **unicast** | **multicast** } **neighbors** **Example:** `switch(config-router-neighbor-af)# show bgp ip unicast neighbors`	Displays the BGP peers.

	Command or Action	Purpose
Step 11	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure the router as a route reflector and add one neighbor as a client:

```
switch(config)# router bgp 65536
switch(config-router)# neighbor 192.0.2.10 remote-as 65536
switch(config-router-neighbor)# address-family ip unicast
switch(config-router-neighbor-af)# route-reflector-client
switch(config-router-neighbor-af)# copy running-config startup-config
```

Configuring Route Dampening

You can configure route dampening to minimize eBGP route flaps propagating through your iBGP network.

To configure route dampening, use the following command in address-family or VRF address family configuration mode:

Command	Purpose
dampening [{ *half-life reuse-limit suppress-limit max-suppress-time* | **route-map** *map-name* }] **Example** : `switch(config-router-af)# dampening route-map bgpDamp`	Disables capabilities negotiation. The parameter values are as follows: • half-life—The range is from 1 to 45. • reuse-limit—The range is from 1 to 20000. • suppress-limit—The range is from 1 to 20000. • max-suppress-time—The range is from 1 to 255.

Configuring Load Sharing and ECMP

You can configure the maximum number of paths that BGP adds to the route table for equal-cost multipath load balancing.

To configure the maximum number of paths, use the following command in router address-family configuration mode:

Command	Purpose
maximum-paths [**ibgp**] *maxpaths* **Example** : `switch(config-router-af)# maximum-paths 12`	Configures the maximum number of equal-cost paths for load sharing. The range is from 1 to 16. The default is 1.

Configuring Maximum Prefixes

You can configure the maximum number of prefixes that BGP can receive from a BGP peer. If the number of prefixes exceeds this value, you can optionally configure BGP to generate a warning message or tear down the BGP session to the peer.

To configure the maximum allowed prefixes for a BGP peer, use the following command in neighbor address-family configuration mode:

Command	Purpose
maximum-prefix *maximum* [*threshold*] [**restart** *time* | **warming-only**] **Example** : `switch(config-router-neighbor-af)# maximum-prefix 12`	Configures the maximum number of prefixes from a peer. The parameter ranges are as follows: • *maximum* —The range is from 1 to 300000. • *Threshold* —The range is from 1 to 100 percent. The default is 75 percent. • *time* —The range is from 1 to 65535 minutes. This command triggers an automatic notification and session reset for the BGP neighbor sessions if the prefix limit is exceeded.

Configuring Dynamic Capability

You can configure dynamic capability for a BGP peer.

To configure dynamic capability, use the following command in neighbor configuration mode:

Command	Purpose
dynamic-capability **Example** : `switch(config-router-neighbor)# dynamic-capability`	Enables dynamic capability. This command triggers an automatic notification and session reset for the BGP neighbor sessions. This command is enabled by default.

Configuring Aggregate Addresses

You can configure aggregate address entries in the BGP route table.

To configure an aggregate address, use the following command in router address-family configuration mode:

Command	Purpose
aggregate-address *ip-prefix/length* [**as-set**] [**summary-only**] [**advertise-map** *map-name*] [**attribute-map** *map-name*] [**suppress-map** *map-name*] **Example** : `switch(config-router-af)# aggregate-address 192.0.2.0/8 as-set`	Creates an aggregate address. The path advertised for this route is an autonomous system set that consists of all elements contained in all paths that are being summarized: • The **as-set** keyword generates autonomous system set path information and community information from contributing paths. • The **summary-only** keyword filters all more specific routes from updates. • The **advertise-map** keyword and argument specify the route map used to select attribute information from selected routes. • The **attribute-map** keyword and argument specify the route map used to select attribute information from the aggregate. • The **suppress-map** keyword and argument conditionally filters more specific routes.

Suppressing BGP Routes

You can configure Cisco NX-OS to advertise newly learned BGP routes only after these routes are confirmed by the Forwarding Information Base (FIB) and programmed in the hardware. After the routes are programmed, subsequent changes to these routes do not require this hardware-programming check. BGP route suppression is not enabled by default.

Note When you enable fib-suppression on the switch for routes that are not programmed locally in the hardware because of hardware table exhaustion, BGP advertises these failed routes even though they are not programmed locally in the hardware.

To suppress BGP routes, use the following command in the router configuration mode:

Command	Purpose
suppress-fib-pending **Example** : `switch(config-router)# suppress-fib-pending`	Suppresses newly learned BGP routes (IPv4 or IPv6) from being advertised to downstream BGP neighbors until the routes have been programmed in the hardware.

Configuring BGP Conditional Advertisement

You can configure BGP conditional advertisement to limit the routes that BGP propagates. You define the following two route maps:

- Advertise map—Specifies the conditions that the route must match before BGP considers the conditional advertisement. This route map can contain any appropriate match statements.

- Exist map or nonexist map—Defines the prefix that must exist in the BGP table before BGP propagates a route that matches the advertise map. The nonexist map defines the prefix that must not exist in the BGP table before BGP propagates a route that matches the advertise map. BGP processes only the permit statements in the prefix list match statements in these route maps.

If the route does not pass the condition, BGP withdraws the route if it exists in the BGP table.

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *as-number*
	Example:
	`switch(config)# router bgp 65536`
	`switch(config-router)#`
		Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

Step 3	**neighbor** *ip-address* **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 192.168.1.2 remote-as 65537`
	`switch(config-router-neighbor)#`
		Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.

Step 4	**address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }
	Example:
	`switch(config-router-neighbor)# address-family ipv4 multicast`
	`switch(config-router-neighbor-af)#`
		Enters address family configuration mode.

Step 5	**advertise-map** *adv-map* { **exist-map** *exist-rmap* | **non-exist-map** *nonexist-rmap* } **Example:** `switch(config-router-neighbor-af)# advertise-map advertise exist-map exist`	Configures BGP to conditionally advertise routes based on the two configured route maps: • *adv-map*—Specifies a route map with **match** statements that the route must pass before BGP passes the route to the next route map. The adv-map is a case-sensitive, alphanumeric string up to 63 characters. • *exist-rmap*—Specifies a route map with **match** statements for a prefix list. A prefix in the BGP table must match a prefix in the prefix list before BGP will advertise

	Command or Action	Purpose
		the route. The exist-rmap is a case-sensitive, alphanumeric string up to 63 characters. • *nonexist-rmap*—Specifies a route map with **match** statements for a prefix list. A prefix in the BGP table must not match a prefix in the prefix list before BGP will advertise the route. The nonexist-rmap is a case-sensitive, alphanumeric string up to 63 characters.
Step 6	(Optional) **show ip bgp neighbor** **Example:** `switch(config-router-neighbor-af)# show ip bgp neighbor`	Displays information about BGP and the configured conditional advertisement route maps.
Step 7	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure BGP conditional advertisement:

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# neighbor 192.0.2.2 remote-as 65537
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# advertise-map advertise exist-map exist
switch(config-router-neighbor-af)# exit
switch(config-router-neighbor)# exit
switch(config-router)# exit
switch(config)# route-map advertise
switch(config-route-map)# match as-path pathList
switch(config-route-map)# exit
switch(config)# route-map exit
switch(config-route-map)# match ip address prefix-list plist
switch(config-route-map)# exit
switch(config)# ip prefix-list plist permit 209.165.201.0/27
```

Configuring Route Redistribution

You can configure BGP to accept routing information from another routing protocol and redistribute that information through the BGP network. Optionally, you can assign a default route for redistributed routes.

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *as-number*
	Example:
	`switch(config)# router bgp 65536`
	`switch(config-router)#`
		Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

Step 3	**address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters address family configuration mode.

Step 4	**redistribute** { **direct** | { **eigrp** | **ospf** | **ospfv3** | **rip** } *instance-tag* | **static** } **route-map** *map-name* **Example:** `switch(config-router-af)# redistribute eigrp 201 route-map Eigrpmap`	Redistributes routes from other protocols into BGP. See the Configuring Route Maps section for more information about route maps.
Step 5	(Optional) **default-metric** *value* **Example:** `switch(config-router-af)# default-metric 33`	Generates a default route into BGP.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-af)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to redistribute EIGRP into BGP:

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# redistribute eigrp 201 route-map Eigrpmap
switch(config-router-af)# copy running-config startup-config
```

Disabling BGP Dampening with Redistribution

When an IGP metric of routes redistributed into BGP changes, BGP has internal dampening that prevents an immediate route update to the BGP peers. It affects how BGP handles IGP metric changes reported for redistributed routes. BGP dampens these changes through a batch process with a 10-minute delay. This command enables you to adjust that delay or remove it altogether for a quicker response to these changes.

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *as-number*
	Example:
	`switch(config)# router bgp 65536`
	`switch(config-router)#`
		Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

Step 3	**address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters address family configuration mode.

Step 4	**dampen-igp-metric** *seconds* **Example:** `switch(config-router-af)# dampen-igp-metric 100`	Configures dampening of IGP metric-related changes for redistributed routes.

Example

This example shows how to configure BGP dampening for redistributed routes:

```
switch# configure terminal
switch(config)# router bgp 100
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# dampen-igp-metric 100
switch(config-router-af)#
```

Configuring Multiprotocol BGP

You can configure MP-BGP to support multiple address families, including IPv4 unicast and multicast routes.

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *as-number*
	Example:
	`switch(config)# router bgp 65536`
	`switch(config-router)#`
		Enters BGP mode and assigns the autonomous system number to the local BGP speaker

Step 3	**neighbor** *ip-address* **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 192.168.1.2 remote-as 65537`
	`switch(config-router-neighbor)#`
		Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.

Step 4	**address-family** { **ipv4** | **ipv6** } { **unicast** | **multicast** }
	Example:
	`switch(config-router-neighbor)# address-family ipv4 multicast`
	`switch(config-router-neighbor-af)#`
		Enters address family configuration mode.

Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to enable advertising and receiving IPv4 routes for multicast RPF for a neighbor:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ipv6 address 2001:0DB8::1
switch(config-if)# router bgp 65536
switch(config-router)# neighbor 192.168.1.2 remote-as 35537
switch(config-router-neighbor)# address-family ipv4 multicast
switch(config-router-neighbor-af)# exit
switch(config-router-neighbor)# address-family ipv6 multicast
switch(config-router-neighbor-af)# copy running-config startup-config
```

Configuring BGP Extended Community Site of Origin

To configure BGP extended community site of origin, use the following commands

Command	Purpose

router bgp *as-number*
Example :
`switch(config)# router bgp 1`
`switch(config-router)#`
	Configures a BGP routing process and enters router configuration mode.

vrf *vrf-name*
Example:
`switch(config-router)# vrf 450`
`switch(config-router-vrf)#`
	Enters the router VRF configuration mode and associates this BGP instance with a VRF.

neighbor ip-address remote-as *as-number*
Example:
`switch(config-router-vrf)# neighbor 1::1 remote-as 2`
`switch(config-router-vrf-neighbor)#`
	Configures the IP address and AS number for a remote BGP peer.

address-family { **ipv4** | **ipv6** } { **multicast** | **unicast** }
Example :
`switch(config-router-vrf-neighbor)# address-family ipv6 unicast`
`switch(config-router-vrf-neighbor-af)#`
	Enters global address family configuration mode for the specified address family.

soo value Example: `switch(config-router-vrf-neighbor-af)# soo 22:14`	Configures the site of origin BGP extended community value. The value is in one of the following formats: • asn:number • IP address:number The number range is from 0 to 65535 for a 2-byte ASN or from 0 to 4294967295 for a 4-byte ASN.

Tuning BGP

You can modify the default behavior of BGP through BGP timers and by adjusting the best-path algorithm.

Configuring Virtualization

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Procedure

	Command or Action	Purpose

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

Step 2	**vrf context** *vrf-name*
	Example:
	`switch(config)# vrf context RemoteOfficeVRF`
	`switch(config-vrf)#`
		Creates a new VRF and enters VRF configuration mode.

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

Step 4	**router bgp** *as-number*
	Example:
	`switch(config)# router bgp 65536`
	`switch(config-router)#`
		Creates a new BGP process with the configured autonomous system number.

Step 5	**vrf** *vrf-name*
	Example:
	`switch(config-router)# vrf RemoteOfficeVRF`
	`switch(config-router-vrf)#`
		Enters the router VRF configuration mode and associates this BGP instance with a VRF.

Step 6	**neighbor** *ip-address* **remote-as** *as-number*
	Example:
	`switch(config-router-vrf)# neighbor 209.165.201.1 remote-as 65536`
	`switch(config-router--vrf-neighbor)#`
		Configures the IP address and AS number for a remote BGP peer.

Step 7	(Optional) **bestpath as-path multipath-relax** **Example:** `switch(config-router-vrf)# bestpath as-path multipath-relax`	Allows the switch to treat paths received from different autonomous systems for multipath, if their autonomous path lengths are the same and other multipath conditions are met.
Step 8	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-neighbor)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to create a VRF and configure the router ID in the VRF:

```
switch# configure terminal
switch(config)# vrf context NewVRF
switch(config-vrf)# exit
switch(config)# router bgp 65536
switch(config-router)# vrf NewVRF
switch(config-router-vrf)# neighbor 209.165.201.1 remote-as 65536
switch(config-router-vrf-neighbor)# copy running-config startup-config
```

Verifying the Advanced BGP Configuration

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

Command	Purpose
show bgp all [**summary**] [**vrf** *vrf-name*]	Displays the BGP information for all address families.
show bgp convergence [**vrf** *vrf-name*]	Displays the BGP information for all address families.
show bgp ip {**unicast**} [*ip-address*] **community** {**regexp** *expression* | [**community**] [**no-advertise**] [**no-export**] [**no-export-subconfed**]} [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP community.
show bgp [**vrf** *vrf-name*] **ip** {**unicast**} [*ip-address*] **community-list** list-name [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP community list.
show bgp ip {**unicast**} [*ip-address*] **extcommunity** {**regexp** *expression* | **generic** [**non-transitive** | **transitive**] *aa4:nn* [**exact-match**]} [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP extended community.
show bgp ip {**unicast**} [ip-address] **extcommunity-list** *list-name* [**exact-match**] [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP extended community list.
show bgp ip {**unicast**} [*ip-address*] {**dampening** *dampened-paths* [**regexp** *expression*]} [**vrf** *vrf-name*]	Displays the information for BGP route dampening. Use the **clear bgp dampening** command to clear the route flap dampening information.
show bgp ip {**unicast**} [*ip-address*] **history-paths** [**regexp** *expression*] [**vrf** *vrf-name*]	Displays the BGP route history paths.
show bgp ip {**unicast**} [*ip-address*] **filter-list** *list-name* [**vrf** *vrf-name*]	Displays the information for the BGP filter list.
show bgp ip {**unicast**} [*ip-address*] **neighbors** [*ip-address*] [**vrf** *vrf-name*]	Displays the information for BGP peers. Use the **clear bgp neighbors** command to clear these neighbors.
show bgp ip {**unicast**} [*ip-address*] {**nexthop** | **nexthop-database**} [**vrf** *vrf-name*]	Displays the information for the BGP route next hop.

Command	Purpose
show bgp paths	Displays the BGP path information.
show bgp ip {**unicast**} [*ip-address*] **policy name** [**vrf** *vrf-name*]	Displays the BGP policy information. Use the **clear bgp policy** command to clear the policy information.
show bgp ip {**unicast**} [*ip-address*] **prefix-list** *list-name* [**vrf** *vrf-name*]	Displays the BGP routes that match the prefix list.
show bgp ip {**unicast**} [*ip-address*] **received-paths** [**vrf** *vrf-name*]	Displays the BGP paths stored for soft reconfiguration.
show bgp ip {**unicast**} [*ip-address*] **regexp** *expression* [**vrf** *vrf-name*]	Displays the BGP routes that match the AS_path regular expression.
show bgp ip {**unicast**} [*ip-address*] **route-map** *map-name* [**vrf** *vrf-name*]	Displays the BGP routes that match the route map.
show bgp peer-policy *name* [**vrf** *vrf-name*]	Displays the information about BGP peer policies.
show bgp peer-session *name* [**vrf** *vrf-name*]	Displays the information about BGP peer sessions.
show bgp peer-template *name* [**vrf** *vrf-name*]	Displays the information about BGP peer templates. Use the **clear bgp peer-template** command to clear all neighbors in a peer template.
show bgp process	Displays the BGP process information.
show ip bgp *options*	Displays the BGP status and configuration information. This command has multiple options. See the Cisco Nexus 3000 Series Command Reference for more information.
show ip mbgp *options*	Displays the BGP status and configuration information. This command has multiple options. See the Cisco Nexus 3000 Series Command Reference for more information.
show running-configuration bgp	Displays the current running BGP configuration.

Displaying BGP Statistics

To display BGP statistics, use the following commands:

Command	Purpose
show bgp ip {**unicast**} [*ip-address*] **flap-statistics** [**vrf** *vrf-name*]	Displays the BGP route flap statistics. Use the **clear bgp flap-statistics** command to clear these statistics.
show bgp sessions [**vrf** *vrf-name*]	Displays the BGP sessions for all peers. Use the **clear bgp sessions** command to clear these statistics.

Command	Purpose
show bgp sessions [**vrf** *vrf-name*]	Displays the BGP sessions for all peers. Use the **clear bgp sessions** command to clear these statistics.
show bgp statistics	Displays the BGP statistics.

Related Topics

The following topics can give more information on BGP:

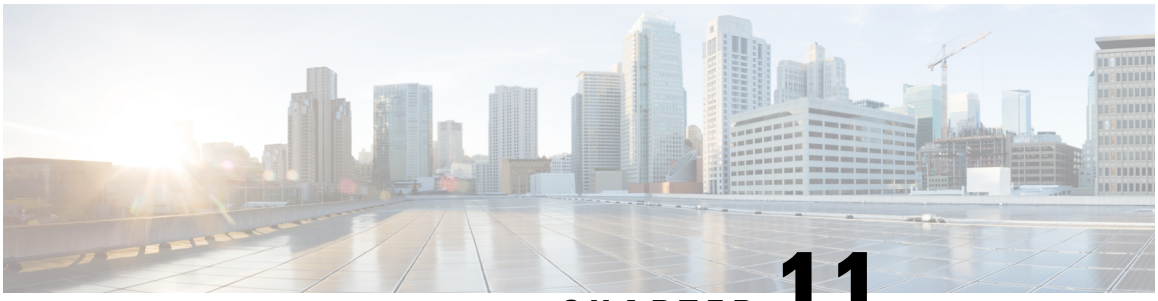
- Configuring Route Policy Manager

Additional References

For additional information related to implementing BGP, see the following sections:

MIBs

MIBs	MIBs Link
BGP4-MIB CISCO-BGP4-MIB	To locate and download MIBs, go to the following: MIB Locator.



CHAPTER 11

Configuring Static Routing

This chapter describes how to configure static route on the router.

This chapter includes the following sections:

- About Static Routing, on page 263
- Prerequisites for Static Routing, on page 265
- Guidelines and Limitations, on page 265
- Default Settings, on page 265
- Configuring Static Routing, on page 265
- Verifying the Static Routing Configuration, on page 267
- Configuration Examples for Static Routing, on page 267

About Static Routing

Routers forward packets using either route information from route table entries that you manually configure or the route information that is calculated using dynamic routing algorithms.

Static routes, which define explicit paths between two routers, cannot be automatically updated; you must manually reconfigure static routes when network changes occur. Static routes use less bandwidth than dynamic routes. No CPU cycles are used to calculate and analyze routing updates.

You can supplement dynamic routes with static routes where appropriate. You can redistribute static routes into dynamic routing algorithms but you cannot redistribute routing information calculated by dynamic routing algorithms into the static routing table.

You should use static routes in environments where network traffic is predictable and where the network design is simple. You should not use static routes in large, constantly changing networks because static routes cannot react to network changes. Most networks use dynamic routes to communicate between routers but may have one or two static routes configured for special cases. Static routes are also useful for specifying a gateway of last resort (a default router to which all unroutable packets are sent).

Administrative Distance

An administrative distance is the metric used by routers to choose the best path when there are two or more routes to the same destination from two different routing protocols. An administrative distance guides the selection of one routing protocol (or static route) over another, when more than one protocol adds the same

route to the unicast routing table. Each routing protocol is prioritized in order of most to least reliable using an administrative distance value.

Static routes have a default administrative distance of 1. A router prefers a static route to a dynamic route because the router considers a route with a low number to be the shortest. If you want a dynamic route to override a static route, you can specify an administrative distance for the static route. For example, if you have two dynamic routes with an administrative distance of 120, you would specify an administrative distance that is greater than 120 for the static route if you want the dynamic route to override the static route.

Directly Connected Static Routes

You need to specify only the output interface (the interface on which all packets are sent to the destination network) in a directly connected static route. The router assumes the destination is directly attached to the output interface and the packet destination is used as the next-hop address. The next-hop can be an interface, only for point-to-point interfaces. For broadcast interfaces, the next-hop must be an IPv4 address.

Fully Specified Static Routes

You must specify the output interface (the interface on which all packets are sent to the destination network) and the next-hop address in a fully specified static route. You can use a fully specified static route when the output interface is a multi-access interface and you need to identify the next-hop address. The next-hop address must be directly attached to the specified output interface.

Floating Static Routes

A floating static route is a static route that the router uses to back up a dynamic route. You must configure a floating static route with a higher administrative distance than the dynamic route that it backs up. In this instance, the router prefers a dynamic route to a floating static route. You can use a floating static route as a replacement if the dynamic route is lost.

Note By default, a router prefers a static route to a dynamic route because a static route has a smaller administrative distance than a dynamic route.

Remote Next-hops for Static Routes

You can specify the next-hop address of a neighboring router that is not directly connected to the router for static routes with remote (non-directly attached) next-hops. If a static route has remote next-hops during data-forwarding, the next-hops are recursively used in the unicast routing table to identify the corresponding directly attached next-hop(s) that have reachability to the remote next-hops

BFD

This feature supports bidirectional forwarding detection (BFD). BFD is a detection protocol designed to provide fast forwarding-path failure detection times. BFD provides subsecond failure detection between two adjacent devices and can be less CPU-intensive than protocol hello messages because some of the BFD load can be distributed onto the data plane on supported modules.

BFD for BGP is supported on eBGP single-hop peers and iBGP single-hop peers. For iBGP single-hop peers using BFD, you must configure the update-source option in neighbor configuration mode. BFD is not supported on other iBGP peers or for multihop eBGP peers.

Virtualization Support

Static routes support virtual routing and forwarding (VRF) instances. By default, Cisco NX-OS places you in the default VRF unless you specifically configure another VRF. For more information, see Configuring Layer 3 Virtualization.

Prerequisites for Static Routing

Static routing has the following prerequisites:

- The next-hop address for a static route must be reachable or the static route will not be added to the unicast routing table.

Guidelines and Limitations

Static routing has the following configuration guidelines and limitations:

- You can specify an interface as the next-hop address for a static route only for point-to-point interfaces such as GRE tunnels.

Default Settings

Following table lists the default settings for static routing parameters.

Table 18: Default Static Routing Parameters

Parameters	Default
Administrative distance	1

Configuring Static Routing

This section includes the following topics:

Configuring a Static Route

You can configure a static route on the router.

Procedure

	Command or Action	Purpose

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

Step 2	**ip route** { *ip-prefix* | *ip-addr ip-mask* } {[*next-hop* | *nh-prefix*] | [*interface next-hop* | *nh-prefix*]} [**name** *nexthop-name*] [**tag** *tag-value* [*pref*] [*track*] [*vrf*] **Example:** `switch(config)# ip route 192.0.2.0/8 ethernet 1/2 192.0.2.4`	Configures a static route and the interface for this static route. You can optionally configure the next-hop address. The *pref* value sets the administrative distance. The range is from 1 to 255. The default is 1.
Step 3	(Optional) **show ip static-route** **Example:** `switch(config)# show ip static-route`	Displays information about static routes.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure a static route:

```
switch# configure terminal
switch(config)# ip route 192.0.2.0/8 192.0.2.10
switch(config)# copy running-config startup-config
```

Use the **no ip static-route** command to remove the static route.

Configuring Virtualization

You can configure a static route in a VRF.

Procedure

	Command or Action	Purpose

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

	Command or Action	Purpose
Step 2	**vrf context** *vrf-name* **Example:** `switch(config)# vrf context StaticVrf`	Configures a static route and the interface for this static route. You can optionally configure the next-hop address. The *pref* value sets the administrative distance. The range is from 1 to 255. The default is 1.
Step 3	**ip route** { *ip-prefix* | *ip-addr ip-mask* } { *next-hop* | *nh-prefix* | *interface* } [**tag** *tag-value* [*pref*]] **Example:** `switch(config-vrf)# ip route 192.0.2.0/8 ethernet 1/2`	Configures a static route and the interface for this static route. You can optionally configure the next-hop address. The *pref* value sets the administrative distance. The range is from 1 to 255. The default is 1.
Step 4	(Optional) **show ip static-route vrf** *vrf-name* **Example:** `switch(config-vrf)# show ip static-route`	Displays information on static routes.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config-vrf)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure a static route:

```
switch# configure terminal
switch(config)# vrf context StaticVrf
switch(config-vrf)# ip route 192.0.2.0/8 192.0.2.10
switch(config-vrf)# copy running-config startup-config
```

Verifying the Static Routing Configuration

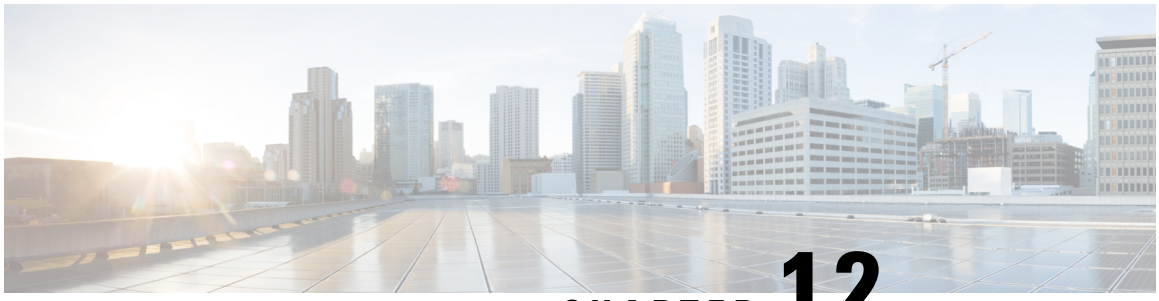
To display the static routing configuration information, use this command:

Command	Purpose
show ip static-route	Displays the configured static routes.

Configuration Examples for Static Routing

This example shows how to configure static routing:

```
configure terminal
ip route 192.0.2.0/8 192.0.2.10
copy running-config startup-config
```



CHAPTER 12

Configuring Layer 3 Virtualization

This chapter describes how to configure Layer 3 virtualization.

This chapter includes the following sections:

- About Layer 3 Virtualization, on page 269
- Guidelines and Limitations for VRF, on page 272
- Guidelines and Limitations for VRF-Lite, on page 272
- Guidelines and Limitations for VRF Route Leaking, on page 273
- Default Settings, on page 273
- Configuring VRFs, on page 274
- Verifying the VRF Configuration, on page 279
- Configuration Examples for VRFs, on page 279

About Layer 3 Virtualization

Overview of Layer 3 Virtualization

Cisco NX-OS supports virtual routing and forwarding instances (VRFs). Each VRF contains a separate address space with unicast and multicast route tables for IPv4 and makes routing decisions independent of any other VRF.

Each router has a default VRF and a management VRF. All Layer 3 interfaces and routing protocols exist in the default VRF until you assign them to another VRF. The mgmt0 interface exists in the management VRF. With the VRF-lite feature, the switch supports multiple VRFs in customer edge (CE) switches. VRF-lite allows a service provider to support two or more virtual private networks (VPNs) with overlapping IP addresses using one interface.

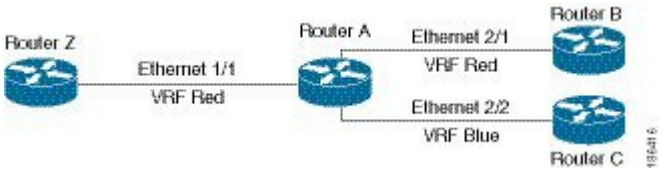
Note The switch does not use Multiprotocol Label Switching (MPLS) to support VPNs.

VRF and Routing

All unicast and multicast routing protocols support VRFs. When you configure a routing protocol in a VRF, you set routing parameters for the VRF that are independent of routing parameters in another VRF for the same routing protocol instance.

You can assign interfaces and route protocols to a VRF to create virtual Layer 3 networks. An interface exists in only one VRF. The following figure shows one physical network split into two virtual networks with two VRFs. Routers Z, A, and B exist in VRF Red and form one address domain. These routers share route updates that do not include router C because router C is configured in a different VRF.

Figure 29: VRFs in a Network



By default, Cisco NX-OS uses the VRF of the incoming interface to select which routing table to use for a route lookup. You can configure a route policy to modify this behavior and set the VRF that Cisco NX-OS uses for incoming packets.

VRF supports route leaking (import or export) between VRFs. Certain limitations apply to route leaking in VRF-Lite. For more information, see Guidelines and Limitations for VRF Route Leaking.

VRF-Lite

VRF-lite is a feature that enables a service provider to support two or more VPNs, where IP addresses can be overlapped among the VPNs. VRF-lite uses input interfaces to distinguish routes for different VPNs and forms virtual packet-forwarding tables by associating one or more Layer 3 interfaces with each VRF. Interfaces in a VRF can be either physical, such as Ethernet ports, or logical, such as VLAN SVIs, but a Layer 3 interface cannot belong to more than one VRF at any time.

Note Multiprotocol Label Switching (MPLS) and MPLS control plane are not supported in the VRF-lite implementation.

Note VRF-lite interfaces must be Layer 3 interfaces.

VRF-Aware Services

A fundamental feature of the Cisco NX-OS architecture is that every IP-based feature is VRF aware.

The following VRF-aware services can select a particular VRF to reach a remote server or to filter information based on the selected VRF:

- AAA

- Call Home
- HSRP
- HTTP
- Licensing
- NTP
- RADIUS
- Ping and Traceroute
- SSH
- SNMP
- Syslog
- TACACS+
- TFTP
- VRRP

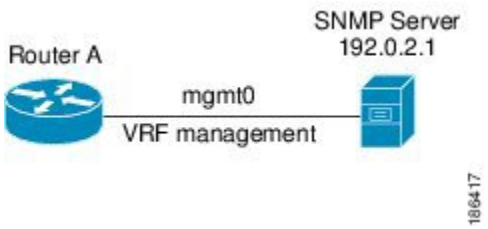
See the appropriate configuration guide for each service for more information on configuring VRF support in that service.

Reachability

Reachability indicates which VRF contains the routing information necessary to get to the server providing the service. For example, you can configure an SNMP server that is reachable on the management VRF. When you configure that server address on the router, you also configure which VRF that Cisco NX-OS must use to reach the server.

The following shows an SNMP server that is reachable over the management VRF. You configure router A to use the management VRF for SNMP server host 192.0.2.1.

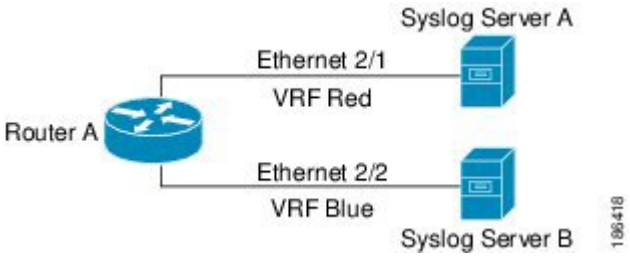
Figure 30: Service VRF Reachability



Filtering

Filtering allows you to limit the type of information that goes to a VRF-aware service based on the VRF. For example, you can configure a syslog server to support a particular VRF. The following figure shows two syslog servers with each server supporting one VRF. syslog server A is configured in VRF Red, so Cisco NX-OS sends only system messages generated in VRF Red to syslog server A.

Figure 31: Service VRF Filtering

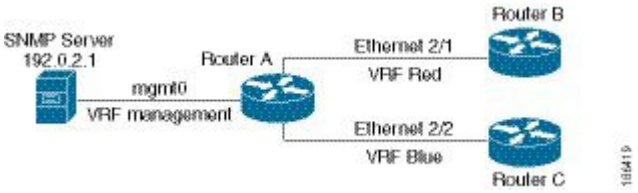


Combining Reachability and Filtering

You can combine reachability and filtering for VRF-aware services. You configure the VRF that Cisco NX-OS uses to connect to that service as well as the VRF that the service supports. If you configure a service in the default VRF, you can optionally configure the service to support all VRFs.

The following figure shows an SNMP server that is reachable on the management VRF. You can configure the SNMP server to support only the SNMP notifications from VRF Red, for example.

Figure 32: Service VRF Reachability Filtering



Guidelines and Limitations for VRF

VRFs have the following configuration guidelines and limitations in a VRF-lite scenario:

- When you make an interface a member of an existing VRF, Cisco NX-OS removes all Layer 3 configuration. You should configure all Layer 3 parameters after adding an interface to a VRF.
- You should add the mgmt0 interface to the management VRF and configure the mgmt0 IP address and other parameters after you add it to the management VRF.
- If you configure an interface for a VRF before the VRF exists, the interface is operationally down until you create the VRF.
- Cisco NX-OS creates the default and management VRFs by default. You should make the mgmt0 interface a member of the management VRF.
- The **write erase boot** command does not remove the management VRF configuration. You must use the **write erase** command and then the **write erase boot** command.

Guidelines and Limitations for VRF-Lite

VRF-lite has the following guidelines and limitations:

- A switch with VRF-lite has a separate IP routing table for each VRF, which is separate from the global routing table.
- Because VRF-lite uses different VRF tables, the same IP addresses can be reused. Overlapped IP addresses are allowed in different VPNs.
- VRF-lite does not support all MPLS-VRF functionality; it does not support label exchange or labeled packets.
- Multiple virtual Layer 3 interfaces can be connected to a VRF-lite switch.
- The switch supports configuring a VRF by using physical ports, VLAN SVIs, or a combination of both. The SVIs can be connected through an access port or a trunk port.
- VRF-lite supports BGP and static routing.
- VRF-lite does not support Enhanced Interior Gateway Routing Protocol (EIGRP).
- VRF-lite does not affect the packet switching rate.
- Multicast cannot be configured on the same Layer 3 interface at the same time.

Guidelines and Limitations for VRF Route Leaking

VRF route leaking has the following guidelines and limitations:

- Route leaking is supported between any two nondefault VRFs. It is also supported between the default VRF and any other VRF.
- Route leaking to the default VRF is not allowed because it is the global VRF.
- You can restrict route leaking to specific routes using route map filters to match designated IP addresses.
- By default, the maximum number of IP prefixes that can be leaked is set to 1000 routes. This number can be configured to any value from 0 to 1000.
- VRF route leaking requires an Enterprise license, and BGP must be enabled.

Default Settings

The following table lists the default settings for VRF parameters.

Table 19: Default VRF Parameters

Parameters	Default
Configured VRFs	Default, management
Routing context	Default VRF

Configuring VRFs

Creating a VRF

You can create a VRF in a switch.

Procedure

	Command or Action	Purpose

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

Step 2	**vrf context name**
	Example:
	`switch(config)# vrf context Enterprise`
	`switch(config-vrf)#`
		Creates a new VRF and enters VRF configuration mode. The **name** can be any case-sensitive, alphanumeric string up to 32 characters.

Step 3	**ip route** { *ip-prefix* | *ip-addr ip-mask* } {[*next-hop* | *nh-prefix*] | [*interface next-hop* | *nh-prefix*]} [**tag** *tag-value* [*pref*] **Example:** `switch(config-vrf)# ip route 192.0.2.0/8 ethernet 1/2 192.0.2.4`	Configures a static route and the interface for this static route. You can optionally configure the next-hop address. The *preference* value sets the administrative distance. The range is from 1 to 255. The default is 1.
Step 4	(Optional) **show vrf** [*vrf-name*] **Example:** `switch(config-vrf)# show vrf Enterprise`	Displays VRF information.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config-vrf)# copy running-config startup-config`	Saves this configuration change.

Example

Use the **no vrf context** command to delete the VRF and the associated configuration:

Command	Purpose
no vrf context *name* **Example:** `switch(config)# no vrf context Enterprise`	Deletes the VRF and all associated configuration.

Any commands available in global configuration mode are also available in VRF configuration mode.

This example shows how to create a VRF and add a static route to the VRF:

```
switch# configure terminal
switch(config)# vrf context Enterprise
switch(config-vrf)# ip route 192.0.2.0/8 ethernet 1/2
switch(config-vrf)# exit
switch(config)# copy running-config startup-config
```

Assigning VRF Membership to an Interface

You can make an interface a member of a VRF.

Before you begin

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

Procedure

	Command or Action	Purpose

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

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

Step 3	**vrf member** *vrf-name* **Example:** `switch(config-if)# vrf member RemoteOfficeVRF`	Adds this interface to a VRF.
Step 4	**ip address** *ip-prefix/length* **Example:** `switch(config-if)# ip address 192.0.2.1/16`	Configures an IP address for this interface. You must do this step after you assign this interface to a VRF.
Step 5	**show vrf** *vrf-name* **interface** *interface-type number* **Example:** `switch(config-if)# show vrf Enterprise interface ethernet 1/2`	Displays VRF information.
Step 6	**copy running-config startup-config** **Example:**	Saves this configuration change.

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

Example

This example shows how to add an interface to the VRF:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# vrf member RemoteOfficeVRF
switch(config-if)# ip address 192.0.2.1/16
switch(config-if)# copy running-config startup-config
```

Configuring VRF Parameters for a Routing Protocol

You can associate a routing protocol with one or more VRFs. See the appropriate chapter for information on how to configure VRFs for the routing protocol. This section uses OSPFv2 as an example protocol for the detailed configuration steps.

Procedure

	Command or Action	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**vrf** *vrf-name*
	Example:
	`switch(config-router)# vrf RemoteOfficeVRF`
	`switch(config-router-vrf)#`
		Enters VRF configuration mode.

Step 4	(Optional) **maximum-paths** *paths* **Example:** `switch(config-router-vrf)# maximum-paths 4`	Configures the maximum number of equal OSPFv2 paths to a destination in the route table for this VRF. Used for load balancing.

Step 5	**interface** *interface-type* **slot/port**
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.

	Command or Action	Purpose
Step 6	**vrf member** *vrf-name* **Example:** `switch(config-if)# vrf member RemoteOfficeVRF`	Adds this interface to a VRF.
Step 7	**ip address** *ip-prefix/length* **Example:** `switch(config-if)# ip address 192.0.2.1/16`	Configures an IP address for this interface. You must do this step after you assign this interface to a VRF.
Step 8	**ip router ospf** *instance-tag* **area** *area-id* **Example:** `switch(config-if)# ip router ospf 201 area 0`	Assigns this interface to the OSPFv2 instance and area configured.
Step 9	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to create a VRF and add an interface to the VRF:

```
switch# configure terminal
switch(config)# vrf context RemoteOfficeVRF
switch(config-vrf)# exit
switch(config)# router ospf 201
switch(config-router)# vrf RemoteOfficeVRF
switch(config-router-vrf)# maximum-paths 4
switch(config-router-vrf)# interface ethernet 1/2
switch(config-if)# vrf member RemoteOfficeVRF
switch(config-if)# ip address 192.0.2.1/16
switch(config-if)# ip router ospf 201 area 0
switch(config-if)# exit
switch(config)# copy running-config startup-config
```

Configuring a VRF-Aware Service

You can configure a VRF-aware service for reachability and filtering. See the VRF-Aware Services section for links to the appropriate chapter or configuration guide for information on how to configure the service for VRFs. This section uses SNMP and IP domain lists as example services for the detailed configuration steps.

Procedure

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

	Command or Action	Purpose
	`switch# configure terminal` `switch(config)#`	
Step 2	**snmp-server host** *ip-address* [**filter-vrf** *vrf-name*] [**use-vrf** *vrf-name*] **Example:** `switch(config)# snmp-server host 192.0.2.1 use-vrf Red`	Configures a global SNMP server and configures the VRF that Cisco NX-OS uses to reach the service. Use the **filter-vrf** keyword to filter information from the selected VRF to this server.

Step 3	**vrf context** *vrf-name*
	Example:
	`switch(config)# vrf context Blue`
	`switch(config-vrf)#`
		Creates a new VRF.

Step 4	**ip domain-list** *domain-name* [**all-vrfs**][**use-vrf** *vrf-name*] **Example:** `switch(config-vrf)# ip domain-list List all-vrfs use-vrf Blue`	Configures the domain list in the VRF and optionally configures the VRF that Cisco NX-OS uses to reach the domain name listed.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config-vrf)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to send SNMP information for all VRFs to SNMP host 192.0.2.1, reachable on VRF Red:

```
switch# configure terminal
switch(config)# snmp-server host 192.0.2.1 for-all-vrfs use-vrf Red
switch(config)# copy running-config startup-config
```

This example shows how to Filter SNMP information for VRF Blue to SNMP host 192.0.2.12, reachable on VRF Red:

```
switch# configure terminal
switch(config)# vrf definition Blue
switch(config-vrf)# snmp-server host 192.0.2.12 use-vrf Red
switch(config)# copy running-config startup-config
```

Setting the VRF Scope

You can set the VRF scope for all EXEC commands (for example, **show** commands). This automatically restricts the scope of the output of EXEC commands to the configured VRF. You can override this scope by using the VRF keywords available for some EXEC commands.

To set the VRF scope, use the following command in EXEC mode:

Command	Purpose
routing-context vrf *vrf-name* **Example:** `switch# routing-context vrf redswitch%red#`	Sets the routing context for all EXEC commands. Default routing context is the default VRF.

To return to the default VRF scope, use the following command in EXEC mode:

Command	Purpose
routing-context vrf default **Example:** `switch# routing-context vrf default`	Sets the default routing context.

Verifying the VRF Configuration

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

Command	Purpose
show vrf [*vrf-name*]	Displays the information for all or one VRF.
show vrf [*vrf-name*] **detail**	Displays detailed information for all or one VRF.
show vrf [*vrf-name*] [**interface** *interface-type slot/port*]	Displays the VRF status for an interface.

Configuration Examples for VRFs

This example shows how to configure VRF Red, add an SNMP server to that VRF, and add an instance of OSPF to VRF Red:

```
vrf context Red
 snmp-server host 192.0.2.12 use-vrf Red
 router ospf 201
 interface ethernet 1/2

vrf member Red
 ip address 192.0.2.1/16
 ip router ospf 201 area 0
```

This example shows how to configure VRF Red and Blue, add an instance of OSPF to each VRF, and create an SNMP context for each OSPF instance in each VRF:

```
vrf context Red
vrf context Blue

feature ospf
 router ospf Lab
 vrf Red
 router ospf Production
 vrf Blue
```

```
interface ethernet 1/2
 vrf member Red
 ip address 192.0.2.1/16
 ip router ospf Lab area 0
 no shutdown

interface ethernet 10/2
 vrf member Blue
 ip address 192.0.2.1/16
 ip router ospf Production area 0
 no shutdown

snmp-server user admin network-admin auth md5 nbv-12345
 snmp-server community public ro
```

Create the SNMP contexts for each VRF

```
snmp-server context lab instance Lab vrf Red
 snmp-server context production instance Production vrf Blue
```

Use the SNMP context **lab** to access the OSPF-MIB values for the OSPF instance Lab in VRF Red in the previous example.

This example shows how to configure route leaking between two non-default VRF's, and from the default VRF to a non-default VRF:

```
feature bgp
 vrf context Green
 ip route 33.33.33.33/32 35.35.1.254
 address-family ipv4 unicast
 route-target import 3:3
 route-target export 2:2
 export map test
 import map test
 import vrf default map test
 interface Ethernet1/7

vrf member Green
 ip address 35.35.1.2/24
 vrf context Shared
 ip route 44.44.44.44/32 45.45.1.254

address-family ipv4 unicast
 route-target import 1:1
 route-target import 2:2
 route-target export 3:3
 export map test
 import map test
 import vrf default map test
 interface Ethernet1/11
 vrf member Shared
 ip address 45.45.1.2/24
 router bgp 100

address-family ipv4 unicast
 redistribute static route-map test
 vrf Green
 address-family ipv4 unicast
 redistribute static route-map test
 vrf Shared
 address-family ipv4 unicast
 redistribute static route-map test
 ip prefix-list test seq 5 permit 0.0.0.0/0 le 32
 route-map test permit 10
 match ip address prefix-list test
```

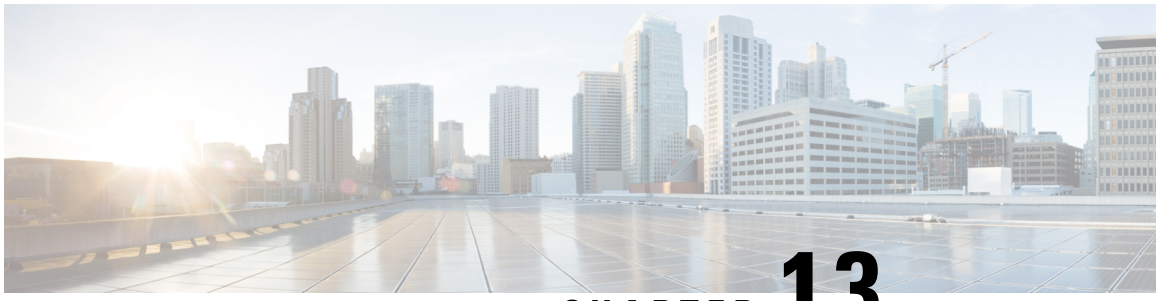
```
 ip route 100.100.100.100/32 55.55.55.1

nexus# show ip route vrf all
 IP Route Table for VRF "default"
 '*' denotes best ucast next-hop
 '**' denotes best mcast next-hop
 '[x/y]' denotes [preference/metric]
 '%<string>' in via output denotes VRF <string
 55.55.55.0/24, ubest/mbest: 1/0, attached
 *via 55.55.55.5, Lo0, [0/0], 00:07:59, direct
 55.55.55.5/32, ubest/mbest: 1/0, attached
 *via 55.55.55.5, Lo0, [0/0], 00:07:59, local
 100.100.100.100/32, ubest/mbest: 1/0
 *via 55.55.55.1, [1/0], 00:07:42, static

IP Route Table for VRF "management"
 '*' denotes best ucast next-hop
 '**' denotes best mcast next-hop
 '[x/y]' denotes [preference/metric]
 '%<string>' in via output denotes VRF <string>
 0.0.0.0/0, ubest/mbest: 1/0
 *via 10.29.176.1, [1/0], 12:53:54, static
 10.29.176.0/24, ubest/mbest: 1/0, attached
 *via 10.29.176.233, mgmt0, [0/0], 13:11:57, direct
 10.29.176.233/32, ubest/mbest: 1/0, attached
 *via 10.29.176.233, mgmt0, [0/0], 13:11:57, local

IP Route Table for VRF "Green"
 '*' denotes best ucast next-hop
 '**' denotes best mcast next-hop
 '[x/y]' denotes [preference/metric]
 '%<string>' in via output denotes VRF <string>
 33.33.33.33/32, ubest/mbest: 1/0
 *via 35.35.1.254, [1/0], 00:23:44, static
 35.35.1.0/24, ubest/mbest: 1/0, attached
 *via 35.35.1.2, Eth1/7, [0/0], 00:26:46, direct
 35.35.1.2/32, ubest/mbest: 1/0, attached
 *via 35.35.1.2, Eth1/7, [0/0], 00:26:46, local
 44.44.44.44/32, ubest/mbest: 1/0
 *via 45.45.1.254%Shared, [20/0], 00:12:08, bgp-100, external, tag 100
 100.100.100.100/32, ubest/mbest: 1/0
 *via 55.55.55.1%default, [20/0], 00:07:41, bgp-100, external, tag 100

IP Route Table for VRF "Shared"
 '*' denotes best ucast next-hop
 '**' denotes best mcast next-hop
 '[x/y]' denotes [preference/metric]
 '%<string>' in via output denotes VRF <string>
 33.33.33.33/32, ubest/mbest: 1/0
 *via 35.35.1.254%Green, [20/0], 00:12:34, bgp-100, external, tag 100
 44.44.44.44/32, ubest/mbest: 1/0
 *via 45.45.1.254, [1/0], 00:23:16, static
 45.45.1.0/24, ubest/mbest: 1/0, attached
 *via 45.45.1.2, Eth1/11, [0/0], 00:25:53, direct
 45.45.1.2/32, ubest/mbest: 1/0, attached
 *via 45.45.1.2, Eth1/11, [0/0], 00:25:53, local
 100.100.100.100/32, ubest/mbest: 1/0
 *via 55.55.55.1%default, [20/0], 00:07:41, bgp-100, external, tag 100
nexus(config)#
```



CHAPTER 13

Configuring the Unicast RIB and FIB

This chapter describes how to configure and manage routes in the unicast Routing Information Base (RIB) and the Forwarding Information Base (FIB) on Cisco NX-OS switches.

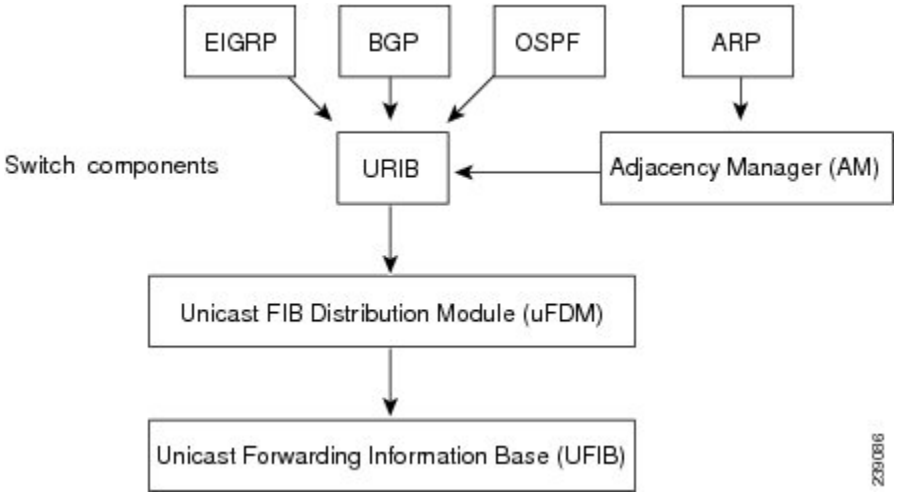
This chapter includes the following sections:

- About the Unicast RIB and FIB, on page 283
- Managing the Unicast RIB and FIB, on page 285
- Verifying the Unicast RIB and FIB Configuration, on page 292

About the Unicast RIB and FIB

The unicast RIB (IPv4 RIB) and FIB are part of the Cisco NX-OS forwarding architecture, as shown in the following figure.

Figure 33: Cisco NX-OS Forwarding Architecture



The unicast RIB maintains the routing table with directly connected routes, static routes, and routes learned from dynamic unicast routing protocols. The unicast RIB also collects adjacency information from sources such as the Address Resolution Protocol (ARP). The unicast RIB determines the best next-hop for a given route and populates the unicast forwarding information base (FIBs) by using the services of the unicast FIB distribution module (FDM).

Each dynamic routing protocol must update the unicast RIB for any route that has timed out. The unicast RIB then deletes that route and recalculates the best next-hop for that route (if an alternate path is available).

Layer 3 Consistency Checker

In rare instances, an inconsistency can occur between the unicast RIB and the FIB on each module. Cisco NX-OS supports the Layer 3 consistency checker. This feature detects inconsistencies between the unicast IPv4 RIB and the FIB on each interface module. Inconsistencies include the following:

- Missing prefix
- Extra prefix
- Wrong next-hop address
- Incorrect Layer 2 rewrite string in the ARP or neighbor discovery (ND) cache

The Layer 3 consistency checker compares the FIB entries to the latest adjacency information from the Adjacency Manager (AM) and logs any inconsistencies. The consistency checker then compares the unicast RIB prefixes to the module FIB and logs any inconsistencies. See the Triggering the Layer 3 Consistency Checker section.

You can then manually clear any inconsistencies. See the Clearing Forwarding Information in the FIB section.

FIB Tables

The following are the unicast routing table capacities for the Cisco Nexus 3500 platform switches when the switch is configured in the normal forwarding mode:

- Unicast Routing Host table = 64,000 hash table entries
- Unicast Routing LPM table = 16,000 TCAM entries
- ECMP members Table size: 16,000 entries

The following are the unicast routing table capacities for the Cisco Nexus 3500 platform switches when the switch is configured inthe warp mode:

- L3 Unicast Host table = 8000 TCAM entries
- L3 Unicast LPM table = 4000 TCAM entries

Note ECMP is not supported on warp mode.

Note In warp mode, when two equal cost paths are received in RIB, one of the paths is installed in the hardware. It is recommended to configure the maximum-path to one under the routing protocol configuration.

Virtualization Support

The Unicast RIB and FIB support Virtual Routing and Forwarding instances (VRFs). For more information, see Configuring Layer 3 Virtualization.

Managing the Unicast RIB and FIB

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

Displaying Module FIB Information

You can display the FIB information on a switch.

DETAILED STEPS

To display the FIB information on a switch, use the following commands in any mode:

Command	Purpose
show ip fib adjacency **Example:** `switch# show ip fib adjacency`	Displays the adjacency information for FIB.
show forwarding ipv4 adjacency **Example:** `switch# show forwarding ipv4 adjacency`	Displays the adjacency information for IPv4.
show ip fib interfaces **Example:** `switch# show ip fib interfaces`	Displays the FIB interface information for IPv4.
show ip fib route **Example:** `switch# show ip fib route`	Displays the route table for IPv4.
show forwarding ipv4 route **Example:** `switch# show forwarding ipv4 route`	Displays the route table for IPv4.

This example shows the FIB contents on a switch:

```
switch# show ip fib route

IPv4 routes for table default/base
```

```
------------------+------------------+--------------------
Prefix | Next-hop | Interface
------------------+------------------+--------------------
0.0.0.0/32 Drop Null0
255.255.255.255/32 Receive sup-eth1
```

Configuring Load Sharing in the Unicast FIB

Dynamic routing protocols, such as Open Shortest Path First (OSPF), support load balancing with equal-cost multipath (ECMP). The routing protocol determines its best routes based on the metrics configured for the protocol and installs up to the protocol-configured maximum paths in the unicast RIB. The unicast RIB compares the administrative distances of all routing protocol paths in the RIB and selects a best path set from all of the path sets installed by the routing protocols. The unicast RIB installs this best path set into the FIB for use by the forwarding plane.

The forwarding plane uses a load-sharing algorithm to select one of the installed paths in the FIB to use for a given data packet.

You can globally configure the following load-sharing settings:

- load-share mode—Selects the best path based on the destination address and port or the source and the destination address and port.
- Universal ID—Sets the random seed for the hash algorithm. You do not need to configure the Universal ID. Cisco NX-OS chooses the Universal ID if you do not configure it.

Load sharing uses the s srkuyo;pame path for all packets in a given flow. A flow is defined by the load-sharing method that you configure. For example, if you configure source-destination load sharing, then all packets with the same source IP address and destination IP address pair follow the same path.

To configure the unicast FIB load-sharing algorithm, use the following command in global configuration mode:

Command	Purpose
ip load-sharing address { **destination port** *destination* | **source-destination** [**port source-destination**]} [**universal-id** *seed*] **Example:** `switch(config)# ip load-sharing address source-destination`	Configures the unicast FIB load-sharing algorithm for data traffic. The *universal-id* range is from 1 to 4294967295.

Configuring Hash Offset

To avoid ECMP polarization in a multi-tier ECMP session, you must configure a different ECMP hash-offset on each tier. Starting with Cisco NX-OS Release 6.0(2)U5(1), a new CLI for ECMP hash concatenation is introduced to achieve uniform distribution of the traffic across 16 way ECMP paths. The updated CLI support exists for Cisco Nexus 3100 platform switches and not on Cisco Nexus 3000 Series switches. You can configure the hash offset in the range of <0-15> in non-concatenate mode and in the range of <0-63> in concatenate mode.

In concatenation mode, if the hash-offset is set to 0 and concatenation is set, the **show running-config** command displays hardware ecmp hash-offset 0 concatenation. The hash-offset is programmed as

per value. On downgrade, if concatenation is configured, the CAP check asks to remove the configuration.

In non-concatenation mode, if the hash-offset is set to 0 and concatenation is reset, the show running-config command does not display hardware ecmp hash-offset 0. The hash-offset is programmed as per value if the hash-offset value is in range 0-15. The hash-offset displays CLI error if the value is in range 16-63 (The non-concatenated mode supports hash-offset for 0-15 range).

The hash-offset in show running-config is visible as per the configured value.

To configure an ECMP hash-offset, use the following commands in global configuration mode:

Command	Purpose
# hardware ecmp hash-offset ?	Configures the ECMP hash-offset. The range is from 0 to 63. The hash offset in the range <0-15> is for the non-concatenate mode. The hash offset in the range <0-63> is for the concatenate mode.
#hardware ecmp hash-offset *number* **Example:** `switch(config)# hardware ecmp hash-offset 5`	Configures the ECMP hash-offset in the non-concatenate mode. The range is from 0 to 15. The default value is 0.
#hardware ecmp hash-offset <0-63> concatenation **Example:** `switch(config)# hardware ecmp hash-offset 63 concatenation`	Configures the ECMP hash-offset in the concatenate mode. **Note** Concatenation support exists for Cisco Nexus 3100 platform switches and not for Cisco Nexus 3000 Series switches. This CLI generates an error on Cisco Nexus 3000 Series switches.

You can use the ECMP hash-offset configured by using the hardware ecmp hash-offset command along with different universal IDs configured by using the ip load-sharing address command to produce various hash results in the load-sharing algorithm.

Configuring Hash Polynomial

Starting with Release 6.0(2)U5(1), new CLI is added for the CRC configuration.

Command	Purpose
`switch# config t`	Enters configuration mode.
`switch(config)# hardware ecmp ? hash-offset Configure hash offsethash-polynomial Configure hash polynomial`	Displays hash-offset and hash-polynomial as the configuration options for hardware ECMP.
`switch config)# hardware ecmp hash-polynomial ? CRC16 Hash polynomial CRC16CRC32HI Hash polynomial CRC32 HI`	Displays CRC16 and CRC32HI as configuration options for Hash polynomial.
`switch config)#show running-config`	Displays the running configuration.

To display the unicast FIB load-sharing algorithm, use the following command in any mode:

Command	Purpose
show ip load-sharing **Example:** `switch(config)# show ip load-sharing`	Displays the unicast FIB load-sharing algorithm for data traffic.

To display the route that the unicast RIB and FIB use for a particular source address and destination address, use the following command in any mode:

Command	Purpose
show routing hash *source-addr dest-addr* [**ip-proto** *ip-protocol*] [*source-l4-port dest-l4-port*] [**vrf** *vrf-name*] **Example:** `switch# show routing hash 1.1.1.6.5.5 5.3 ip-proto 0x11 10 234`	Displays the route that the unicast RIB FIB use for a source and destination address pair. The source address and destination address format is x.x.x.x. The source port and destination port range is from 1 to 65535. The VRF name can be any case-sensitive, alphanumeric string up to 64 characters. The ip-proto option corresponds to the protocol field of the IP header.

This example shows how to display the route selected for a source/destination pair:

```
switch# show routing hash 1.1.1.6.5.5.5.3 ip-proto 0x11 10 234
Load-share parameters used for software forwarding:
load-share mode: address source-destination port source-destination
Universal-id seed: 0xe05e2e85
Invoking pc_ic_ecmp_resolution
Hash for VRF "default"
Hashing to path *Eth1/29%
For route:
5.5.5.0/24 ubest/mbest: 3/0
*via 2.2.2.1, Eth1/18, [1/0], 00:14:14, static
*via 3.3.3.1, Eth1/29, [1/0], 00:14:14, static
*via 4.4.4.1, Eth1/34, [1/0], 00:14:14, static
```

Displaying Routing and Adjacency Information

ou can display the routing and adjacency information.

To display the routing and adjacency information, use the following commands in any mode:

Command	Purpose
show ip route [**route-type** | **interface int-type number** | **next-hop**] **Example:** `switch# show ip route`	Displays the unicast route table. The *route-type* argument can be a single route prefix, direct, static, or a dynamic route protocol. Use the **?** keyword to see the supported interfaces.

Command	Purpose
show ip adjacency [*prefix* | *interface number* [**summary**] | **non-best**] [**detail**] [**vrf** *vrf-id*] **Example:** `switch# show ip adjacency`	Displays the adjacency table. The argument ranges are as follows: • *prefix* —Any IPv4prefix address. • *interface-type number* —Use the **?** keyword to see the supported interfaces. • *vrf-id* —Any case-sensitive, alphanumeric string up to 32 characters.
show ip routing [*route-type* | **interface** *int-type number* | **next-hop** | **recursive-next-hop** | **summary** | **updated** { **since** | **until** } *time*] **Example:** `switch# show routing summary`	Displays the unicast route table. The *route-type* argument can be a single route prefix, direct, static, or a dynamic route protocol. Use the **?** keyword to see the supported interfaces.

This example displays the unicast route table:

```
switch# show ip route
IP Route Table for VRF "default"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]

192.168.0.2/24, ubest/mbest: 1/0, attached
*via 192.168.0.32, Eth1/5, [0/0], 22:34:09, direct
192.168.0.32/32, ubest/mbest: 1/0, attached
*via 192.168.0.32, Eth1/5, [0/0], 22:34:09, local
```

This example shows the adjacency information:

```
switch# show ip adjacency

IP Adjacency Table for VRF default
Total number of entries: 2
Address Age MAC Address Pref Source Interface Best
10.1.1.1 02:20:54 00e0.b06a.71eb 50 arp mgmt0 Yes
10.1.1.253 00:06:27 0014.5e0b.81d1 50 arp mgmt0 Yes
```

Triggering the Layer 3 Consistency Checker

You can manually trigger the Layer 3 consistency checker.

To manually trigger the Layer 3 consistency checker, use the following commands in global configuration mode:

Command	Purpose
test [**ipv4**] [**unicast**] **forwarding inconsistency** [**vrf** *vrf-name*] [**module** { *slot* | **all** }] **Example:** `switch(config)# test forwarding inconsistency`	Starts a Layer 3 consistency check. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters. The *slot* range is from 1 to 10.

To stop the Layer 3 consistency checker, use the following commands in global configuration mode:

Command	Purpose
test forwarding [**ipv4**] [**unicast**] **inconsistency** [**vrf** *vrf-name*] [**module** { *slot* | **all** }] **stop** **Example:** `switch(config)# test forwarding inconsistency stop`	Stops a Layer 3 consistency check. The *vrf-name* can be any case-sensitive, alphanumeric string up to 64 characters. The *slot* range is from 1 to 10.

To display the Layer 3 inconsistencies, use the following commands in any mode:

Command	Purpose
show forwarding [**ipv4**] **inconsistency** [**vrf** *vrf-name*] [**module** { *slot* | **all** }] **Example:** `switch(config)# show forwarding inconsistency`	Displays the results of a Layer 3 consistency check. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters. The *slot* range is from 1 to 10.

Clearing Forwarding Information in the FIB

You can clear one or more entries in the FIB.

Note The **clear forwarding** command disrupts forwarding on the switch.

To clear an entry in the FIB, including a Layer 3 inconsistency, use the following command in any mode:

Command	Purpose
clear forwarding { **ipv4** } **route** {* | *prefix* } [**vrf** *vrf-name*] [**module** { *slot* | **all** }] **Example:** `switch(config)# clear forwarding ipv4 route *`	Clears one or more entries from the FIB. The route options are as follows: • *—All routes. • *prefix* —Any IPprefix. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters. The **slot** range is from 1 to 10.

Note Ensure you clear the RIB entry after you clear the FIB entry.

Estimating Memory Requirements for Routes

You can estimate the memory that a number of routes and next-hop addresses will use.

To estimate the memory requirements for routes, use the following command in any mode:

Command	Purpose
show routing memory estimate routes *num-routes* **next-hops** *num-nexthops* **Example:** `switch# show routing memory estimate routes 1000 next-hops 1`	Displays the memory requirements for routes. The *num-routes* range is from 1000 to 1000000. The *num-nexthops* range is from 1 to 16.

Clearing Routes in the Unicast RIB

You can clear one or more routes from the unicast RIB.

Caution The * keyword is severely disruptive to routing.

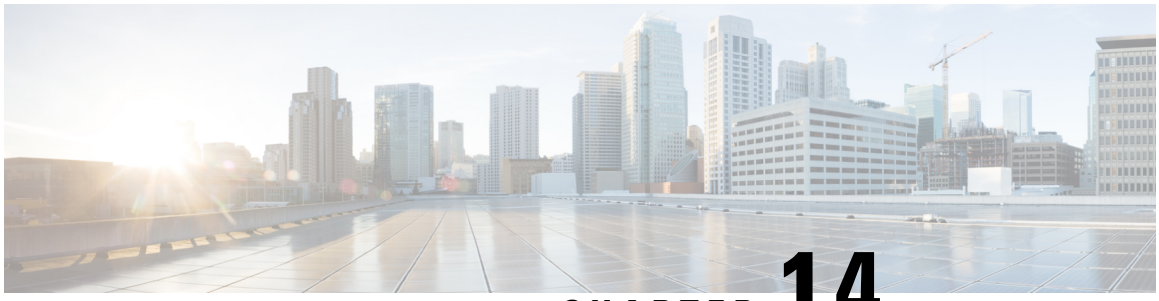
To clear one or more entries in the unicast RIB, use the following commands in any mode:

Command	Purpose
clear iproute {* | { *route* | *prefix/length* }[*next-hop interface*]} [**vrf** *vrf-name*] **Example:** `switch(config)# clear ip route 10.2.2.2`	Clears one or more routes from both the unicast RIB and all the module FIBs. The route options are as follows: • *—All routes. • *route* —An individual IProute. • *prefix/length* —Any IPprefix. • *next-hop* —The next-hop address • *interface* —The interface to reach the next-hop address. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
clear routing unicast [**ip** | **ipv4**] {* | { *route* | *prefix/length* }[*next-hop interface*]} [**vrf** *vrf-name*] **Example:** `switch(config)# clear routing ip 10.2.2.2`	Clears one or more routes from the unicast RIB. The route options are as follows: • *—All routes. • *route* —An individual IProute. • *prefix/length* —Any IPprefix. • *next-hop* —The next-hop address • *interface* —The interface to reach the next-hop address. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.

Verifying the Unicast RIB and FIB Configuration

To display the unicast RIB and FIB configuration information, perform one of the following tasks:

Command	Purpose
show forwarding adjacency	Displays the adjacency table on a module.
show forwarding distribution { **clients** | **fib-state** }	Displays the FIB distribution information.
show forwarding interfaces module *slot*	Displays the FIB information for a module.
show forwarding ipv4route	Displays routes in the FIB.
show ip adjacency	Displays the adjacency table.
show ip route	Displays IPv4routes from the unicast RIB.
show routing	Displays routes from the unicast RIB.



CHAPTER 14

Configuring Route Policy Manager

This chapter describes how to configure the Route Policy Manager on Cisco NX-OS switches.

This chapter includes the following sections:

- About Route Policy Manager, on page 293
- Guidelines and Limitations for Route Policy Manager, on page 297
- Default Settings, on page 298
- Configuring Route Policy Manager, on page 298
- Verifying the Route Policy Manager Configuration, on page 308
- Related Topics, on page 308

About Route Policy Manager

Route Policy Manager supports route maps and IP prefix lists. These features are used for route redistribution. A prefix list contains one or more IPv4 network prefixes and the associated prefix length values. You can use a prefix list by itself in features such as Border Gateway Protocol (BGP) templates, route filtering, or redistribution of routes that are exchanged between routing domains.

Route maps can apply to both routes and IP packets. Route filtering and redistribution pass a route through a route map.

Prefix Lists

You can use prefix lists to permit or deny an address or range of addresses. Filtering by a prefix list involves matching the prefixes of routes or packets with the prefixes listed in the prefix list. An implicit deny is assumed if a given prefix does not match any entries in a prefix list.

You can configure multiple entries in a prefix list and permit or deny the prefixes that match the entry. Each entry has an associated sequence number that you can configure. If you do not configure a sequence number, Cisco NX-OS assigns a sequence number automatically. Cisco NX-OS evaluates prefix lists starting with the lowest sequence number. Cisco NX-OS processes the first successful match for a given prefix. Once a match occurs, Cisco NX-OS processes the permit or deny statement and does not evaluate the rest of the prefix list.

Note An empty prefix list permits all routes.

Prefix List Masks

Masks can be used for prefix lists. Masking uses the number 1 and the number 0 to specify how the software treats the corresponding IP address bits.

- A mask bit 0 means ignore the corresponding bit value.
- A mask bit 1 means check the corresponding bit value for an exact match.

You can use a prefix list to match the IP address in a route-map, which in turn is used in routing protocols during redistribution. The IP address is matched against the prefix list where the bits corresponding to the mask bit 1 are the same as the subnet provided in the prefix list.

By carefully setting masks, you can select a single, or several IP addresses for permit or deny tests.

The prefix list mask allows noncontiguous bits in the mask. You can thus define a range of even or odd numbered IP addresses.

MAC Lists

You can use MAC lists to permit or deny MAC address or range of addresses. A MAC list consists of a list of MAC addresses and optional MAC masks. A MAC mask is a wildcard mask that is logically AND-ed with the MAC address when the route map matches on the MAC list entry. Filtering by a MAC list involves matching the MAC address of packets with the MAC addresses listed in the MAC list. An implicit deny is assumed if a given MAC address does not match any entries in a MAC list.

You can configure multiple entries in a MAC list and permit or deny the MAC addresses that match the entry. Each entry has an associated sequence number that you can configure. If you do not configure a sequence number, Cisco NX-OS assigns a sequence number automatically. Cisco NX-OS evaluates MAC lists starting with the lowest sequence number. Cisco NX-OS processes the first successful match for a given MAC address. Once a match occurs, Cisco NX-OS processes the permit or deny statement and does not evaluate the rest of the MAC list.

Route Maps

You can use route maps for route redistribution. Route map entries consist of a list of match and set criteria. The match criteria specify match conditions for incoming routes or packets, and the set criteria specify the action taken if the match criteria are met.

You can configure multiple entries in the same route map. These entries contain the same route map name and are differentiated by a sequence number.

You create a route map with one or more route map entries arranged by the sequence number under a unique route map name. The route map entry has the following parameters:

- Sequence number
- Permission—permit or deny
- Match criteria
- Set changes

By default, a route map processes routes or IP packets in a linear fashion, that is, starting from the lowest sequence number. You can configure the route map to process in a different order using the **continue** statement, which allows you to determine which route map entry to process next.

Match Criteria

You can use a variety of criteria to match a route or IP packet in a route map. Some criteria, such as BGP community lists, are applicable only to a specific routing protocol, while other criteria, such as the IP source or the destination address, can be used for any route or IP packet.

When Cisco NX-OS processes a route or packet through a route map, it compares the route or packet to each of the match statements configured. If the route or packet matches the configured criteria, Cisco NX-OS processes it based on the permit or deny configuration for that match entry in the route map and any set criteria configured.

The match categories and parameters are as follows:

- BGP parameters—Match based on AS numbers, AS-path, community attributes, or extended community attributes.
- Prefix lists—Match based on an address or range of addresses.
- Multicast parameters—Match based on a rendezvous point, groups, or sources.
- Other parameters—Match based on IP next-hop address or packet length.

Set Changes

Once a route or packet matches an entry in a route map, the route or packet can be changed based on one or more configured set statements.

The set changes are as follows:

- BGP parameters—Change the AS-path, tag, community, extended community, dampening, local preference, origin, or weight attributes.
- Metrics—Change the route-metric, the route-tag, or the route-type.
- Other parameters—Change the forwarding address or the IP next-hop address.

Access Lists

IP access lists can match the packet to a number of IP packet fields such as the following:

- Source or destination IPv4 address
- Protocol
- Precedence
- ToS

AS Numbers for BGP

You can configure a list of AS numbers to match against BGP peers. If a BGP peer matches an AS number in the list and matches the other BGP peer configuration, BGP creates a session. If the BGP peer does not

match an AS number in the list, BGP ignores the peer. You can configure the AS numbers as a list, a range of AS numbers, or you can use an AS-path list to compare the AS numbers against a regular expression.

AS-Path Lists for BGP

You can configure an AS-path list to filter inbound or outbound BGP route updates. If the route update contains an AS-path attribute that matches an entry in the AS-path list, the router processes the route based on the permit or deny condition configured. You can configure AS-path lists within a route map.

You can configure multiple AS-path entries in an AS-path list by using the same AS-path list name. The router processes the first entry that matches.

Community Lists for BGP

You can filter BGP route updates based on the BGP community attribute by using community lists in a route map. You can match the community attribute based on a community list, and you can set the community attribute using a route map.

A community list contains one or more community attributes. If you configure more than one community attribute in the same community list entry, then the BGP route must match all community attributes listed to be considered a match.

You can also configure multiple community attributes as individual entries in the community list by using the same community list name. In this case, the router processes the first community attribute that matches the BGP route, using the permit or deny configuration for that entry.

You can configure community attributes in the community list in one of the following formats:

- A named community attribute, such as **internet** or **no-export**.
- In *aa:nn* format, where the first two bytes represent the two-byte AS number and the last two bytes represent a user-defined network number.
- A regular expression.

Extended Community Lists for BGP

Extended community lists support 4-byte AS numbers. You can configure community attributes in the extended community list in one of the following formats:

- In *aa4:nn* format, where the first four bytes represent the four-byte AS number and the last two bytes represent a a user-defined network number.
- A regular expression.

Cisco NX-OS supports generic-specific extended community lists, which provide similar functionality to regular community lists for four-byte AS numbers. You can configure generic-specific extended community lists with the following properties:

- Transitive—BGP propagates the community attributes across autonomous systems.
- Nontransitive—BGP removes community attributes before propagating the route to another autonomous system.

Route Redistribution and Route Maps

You can use route maps to control the redistribution of routes between routing domains. Route maps match on the attributes of the routes to redistribute only those routes that pass the match criteria. The route map can also modify the route attributes during this redistribution using the set changes.

The router matches redistributed routes against each route map entry. If there are multiple match statements, the route must pass all of the match criteria. If a route passes the match criteria defined in a route map entry, the actions defined in the entry are executed. If the route does not match the criteria, the router compares the route against subsequent route map entries. Route processing continues until a match is made or the route is processed by all entries in the route map with no match. If the router processes the route against all entries in a route map with no match, the router accepts the route (inbound route maps) or forwards the route (outbound route maps).

Guidelines and Limitations for Route Policy Manager

Route Policy Manager has the following configuration guidelines and limitations:

- Although CLI allows **set** or **match** on **route-tag**, it is not supported and will cause unintended behavior for that particular route-map sequence.
- Names in the prefix-list are case-insensitive. We recommend using unique names. Do not use the same name by modifying upper-case and lower-case characters. For example, CTCPrimaryNetworks and CtcPrimaryNetworks are two different entries.
- If no route map exists, all routes are denied.
- If no prefix list exists, all routes are permitted.
- Without any match statement in a route-map entry, the permission (permit or deny) of the route-map entry decides the result for all the routes or packets.
- If referred policies (for example, prefix lists) within a match statement of a route-map entry return either a no-match or a deny-match, Cisco NX-OS fails the match statement and processes the next route-map entry.
- When you change a route map, Cisco NX-OS holds all the changes until you exit from the route-map configuration submode. Cisco NX-OS then sends all the changes to the protocol clients to take effect.
- Cisco recommends that you do not have both IPv4 and IPv6 match statements in the same route-map sequence. If both are required, they should be specified in different sequences in the same route-map.
- Because you can use a route map before you define it, verify that all your route maps exist when you finish a configuration change.
- You can view the route-map usage for redistribution and filtering. Each individual routing protocol provides a way to display these statistics.
- When you redistribute BGP to IGP, iBGP is redistributed as well. To override this behavior, you must insert an additional deny statement into the route map.
- Route Policy Manager does not support MAC lists.
- The maximum number of characters for ACL names in the ip access-list name command is 64. However, ACL names that are associated with RPM commands (such as ip prefix-list and match ip address) accept a maximum of only 63 characters.

- BGP supports only specific **match** commands. For details, see the **match** commands table in the Configuring Route Maps, on page 303 section.
- If you create an ACL named "prefix-list," it cannot be associated with a route map that is created using the match ip address command. The RPM command match ip address prefix-list makes the previous command (with the "prefix-list" ACL name) ambiguous.
- You can configure only one ACL when using the match ip address command.

Default Settings

Following lists the default settings for Route Policy Manager.

Table 20: Default Route Policy Manager Parameters

Parameters	Default
Route Policy Manager	Enabled

Configuring Route Policy Manager

Configuring IP Prefix Lists

IP prefix lists match the IP packet or route against a list of prefixes and prefix lengths. You can create an IP prefix list for IPv4.

You can configure the prefix list entry to match the prefix length exactly, or to match any prefix with a length that matches the configured range of prefix lengths.

Use the **ge** and **lt** keywords to create a range of possible prefix lengths. The incoming packet or route matches the prefix list if the prefix matches and if the prefix length is greater than or equal to the **ge** keyword value (if configured) and less than or equal to the **lt** keyword value (if configured). When using the **eq** keyword, the value you set must be greater than the mask length for the prefix.

Use the **mask** keyword to define a range of possible contiguous or non-contiguous routes to be compared to the prefix address.

Procedure

	Command or Action	Purpose

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

Step 2	(Optional) **ip prefix-list** *name* **description** *string*	Adds an information string about the prefix list.

	Command or Action	Purpose
	Example: `switch(config)# ip prefix-list AllowPrefix description allows engineering server`	

Step 3	**ip prefix-list** *name* [**seq** *number*] [{ **permit** | **deny** } *prefix* {[**eq** *prefix-length*] | [**ge** *prefix-length*] [**le** *prefix-length*]}] [**mask** *mask*]
	Example:
	`switch(config)# ip prefix-list AllowPrefix seq 10 permit 192.0.2.0/23 eq 24`
	`switch(config)# ip prefix-list even permit 0.0.0.0/32 mask 0.0.0.1`
		Creates an IPv4 prefix list or adds a prefix to an existing prefix list. The prefix length is matched as follows:
		• eq—Matches the exact *prefix length*. This value must be greater than the mask length.
		• ge—Matches a prefix length that is equal to or greater than the configured *prefix length*.
		• le—Matches a prefix length that is equal to or less than the configured *prefix length*.
		• mask—Specifies the bits of a prefix address in a prefix list that are compared to the bits of the prefix address used in routing protocols during redistribution.

Step 4	(Optional) **show ip prefix-list** *name* **Example:** `switch(config)# show ip prefix-list AllowPrefix`	Displays information about prefix lists.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to create an IPv4 prefix list with two entries and apply the prefix list to a BGP neighbor:

```
switch# configure terminal
switch(config)# ip prefix-list allowprefix seq 10 permit 192.0.2.0/23 eq 24
switch(config)# ip prefix-list allowprefix seq 20 permit 209.165.201.0/27 eq 28
switch(config)# router bgp 65536:20
switch(config-router)# neighbor 192.0.2.1/16 remote-as 65535:20
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# prefix-list allowprefix in
```

This example shows how to create an IPv4 prefix list with a match mask for odd IP addresses:

```
switch# configure terminal
switch(config)# ip prefix-list odd permit 0.0.0.1/32 mask 0.0.0.1
```

Configuring MAC Lists

You can configure a MAC list to permit or deny a range of MAC addresses.

Procedure

	Command or Action	Purpose

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

Step 2	**mac-list** *name* [**seq** *number*] { } mac-address {mac-mask] **Example:** `switch(config)# mac-list AllowMac seq 1 permit 0022.5579.a4c1 ffff.ffff.0000`	Creates a MAC list or adds a MAC address to an existing MAC list. The *seq* range is from 1 to 4294967294. The *mac-mask* specifies the portion of the MAC address to match against and is in MAC address format.
Step 3	(Optional) **show mac- list** *name* **Example:** `switch(config)# show mac-list AllowMac`	Displays information about MAC lists.
Step 4	(Optional) copy running-config startup-config **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Configuring AS-Path Lists

You can specify an AS-path list filter on both inbound and outbound BGP routes. Each filter is an access list based on regular expressions. If the regular expression matches the representation of the AS-path attribute of the route as an ASCII string, then the permit or deny condition applies.

Procedure

	Command or Action	Purpose

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

Step 2	**ip as-path access-list** *name* { **deny** | **permit** } *expression* **Example:** `switch(config)# ip as-path access-list Allow40 permit 40`	Creates a BGP AS-path list using a regular expression.

	Command or Action	Purpose
Step 3	(Optional) **show ip as-path-access-list** *name* **Example:** `switch(config)# show ip as-path-access-list Allow40`	Displays information about as-path access lists.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to create an AS-path list with two entries and apply the AS path list to a BGP neighbor:

```
switch# configure terminal
switch(config)# ip as-path access-list AllowAS permit 64510
switch(config)# ip as-path access-list AllowAS permit 64496
switch(config)# copy running-config startup-config
switch(config)# router bgp 65536:20
switch(config-router)# neighbor 192.0.2.1/16 remote-as 65535:20
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# filter-list AllowAS in
```

Configuring Community Lists

You can use community lists to filter BGP routes based on the community attribute. The community number consists of a 4-byte value in the *aa:nn* format. The first two bytes represent the autonomous system number, and the last two bytes represent a user-defined network number.

When you configure multiple values in the same community list statement, all community values must match to satisfy the community list filter. When you configure multiple values in separate community list statements, the first list that matches a condition is processed.

Use community lists in a match statement to filter BGP routes based on the community attribute.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal switch(config)#`	Enters global configuration mode.
Step 2	<table><tr><th>Option</th><th>Description</th></tr><tr><td>Command</td><td>Description</td></tr></table>	

<table>
<tr><th></th><th colspan="2">Command or Action</th><th>Purpose</th></tr>
<tr><td></td><td colspan="2"><table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td>ip community-list standard list-name { deny | permit } [community-list] [internet] [local-AS] [no-advertise] [no-export]
Example:
<code>switch(config)# ip community-list standard BGPCommunity permit no-advertise 65536:20</code></td><td>Creates a standard BGP community list. The list-name can be any case-sensitive, alphanumeric string up to 63 characters. The community-list can be one or more communities in the aa:nn format.</td></tr>
<tr><td>ip community-list expanded list-name { deny | permit } expression
Example:
<code>switch(config)# ip community-list expanded BGPComplex deny 50000:[0-9][0-9]_</code></td><td>Creates an expanded BGP community list using a regular expression.</td></tr>
</table></td><td></td></tr>
<tr><td>Step 3</td><td colspan="2">(Optional) show ip community-list name
Example:
<code>switch(config)# show ip community-list BGPCommunity</code></td><td>Displays information about community lists.</td></tr>
<tr><td>Step 4</td><td colspan="2">(Optional) copy running-config startup-config
Example:
<code>switch(config)# copy running-config startup-config</code></td><td>Saves this configuration change.</td></tr>
</table>

Example

This example shows how to create a community list with two entries:

```
switch# configure terminal
switch(config)# ip community-list standard BGPCommunity permit no-advertise 65536:20
switch(config)# ip community-list standard BGPCommunity permit local-AS no-export
switch(config)# copy running-config startup-config
```

Configuring Route Maps

You can use route maps for route redistribution or route filtering. Route maps can contain multiple match criteria and multiple set criteria.

Configuring a route map for BGP triggers an automatic soft clear or refresh of BGP neighbor sessions.

Procedure

	Command or Action	Purpose

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

Step 2	**route-map** *map-name* [**permit** | **deny**] [*seq*]
	Example:
	`switch(config)# route-map Testmap permit 10`
	`switch(config-route-map)#`
		Creates a route map or enters route-map configuration mode for an existing route map. Use *seq* to order the entries in a route map.

Step 3	(Optional) **continue** *seq* **Example:** `switch(config-route-map)# continue 10`	Determines what sequence statement to process next in the route map. Used only for filtering and redistribution.
Step 4	(Optional) **exit** **Example:** `switch(config-route-map)# exit`	Exits route-map configuration mode.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	Saves this configuration change.

Example

You can configure the following optional match parameters for route maps in route-map configuration mode:

Note The **default-information originate** command ignores **match** statements in the optional route map.

Command	Purpose
match as-path *name* [*name...*] **Example:** `switch(config-route-map)# match as-path Allow40`	Matches against one or more AS-path lists. Create the AS-path list with the **ip as-path access-list** command.
match as-number { *number* [,*number*...] | **as-path-list** *name* [*name*...]} **Example:** `switch(config-route-map)# match as-number 33,50-60`	Matches against one or more AS numbers or AS-path lists. Create the AS-path list with the **ip as-path access-list** command. The number range is from 1 to 65535. The AS-path list name can be any case-sensitive, alphanumeric string up to 63 characters.
match community name [*name*...] [**exact-match**] **Example:** `switch(config-route-map)# match community BGPCommunity`	Matches against one or more community lists. Create the community list with the **ip community-list** command.
match extcommunity *name* [*name*...] [**exact-match**] **Example:** `switch(config-route-map)# match extcommunity BGPextCommunity`	Matches against one or more extended community lists. Create the community list with the **ip extcommunity-list** command.
match interface *interface-type number* [*interface-type number...*] **Example:** `switch(config-route-map)# match interface e 1/2`	Matches any routes that have their next hop out one of the configured interfaces. Use **?** to find a list of supported interface types.
match ip address prefix-list *name* [*name...*] **Example:** `switch(config-route-map)# match ip address prefix-list AllowPrefix`	Matches against one or more IPv4 prefix lists. Use the **ip prefix-list** command to create the prefix list.
match ip next-hop prefix-list *name* [*name* ...] **Example:** `switch(config-route-map)# match ip next-hop prefix-list AllowPrefix`	Matches the IPv4 next-hop address of a route to one or more IP prefix lists. Use the *ip prefix-list* command to create the prefix list.
match ip route-source prefix-list *name* [*name* ...] **Example:** `switch(config-route-map)# match ip route-source prefix-list AllowPrefix`	Matches the IPv4 route source address of a route to one or more IP prefix lists. Use the **ip prefix-list** command to create the prefix list.

Command	Purpose
match mac-list *name* [*name...*] **Example:** `switch(config-route-map)# match mac-list AllowMAC`	Matches against one or more MAC lists. Use the **mac-list** command to create the MAC list.
match metric *value* [*+-deviation*] [*value..*] **Example:** `switch(config-route-map)# match mac-list AllowMAC`	Matches the route metric against one or more metric values or value ranges. Use +- **deviation** argument to set a metric range. The route map matches any route metric that falls the range: *value - deviation* to *value + deviation*.
match route-type *route-type* **Example:** `switch(config-route-map)# match route-type level 1 level 2`	Matches against a type of route. The *route-type* can be one or more of the following: • external • internal • level-1 • level-2 • local • nssa-external • type-1 • type-2
match tag *tagid* [*tagid...*] **Example:** `switch(config-route-map)# match tag 2`	Matches a route against one or more tags for filtering or redistribution.
match vlan *vlan-id* [*vlan-range*] **Example:** `switch(config-route-map)# match vlan 3, 5-10`	Matches against a VLAN.

You can configure the following optional set parameters for route maps in route-map configuration mode:

Command	Purpose
set as-path { **tag** | **prepend** { **last-as** *number* | *as-1* [*as-2...*]}} **Example:** `switch(config-route-map)# set as-path prepend 10 100 110`	Modifies an AS-path attribute for a BGP route. You can prepend the configured*number* of last AS numbers or a string of particular AS-path values (*as-1 as-2...as-n*).

Command	Purpose
set comm-list *name* **delete** **Example:** `switch(config-route-map)# set comm-list BGPCommunity delete`	Removes communities from the community attribute of an inbound or outbound BGP route update. Use the **ip community-list** command to create the community list.
set community { **none** | **additive** | **local-AS** | **no-advertise** | **no-export** | *community-1* [*community-2...*]} **Example:** `switch(config-route-map)# set community local-AS`	Sets the community attribute for a BGP route update. **Note** When you use both the **set community** and **set comm-list delete** commands in the same sequence of a route map attribute, the deletion operation is performed before the set operation. **Note** Use the **send-community** command in BGP neighbor address family configuration mode to propagate BGP community attributes to BGP peers.
set dampening *halflife reuse suppress duration* **Example:** `switch(config-route-map)# set dampening 30 1500 10000 120`	Sets the following BGP route dampening parameters: • *halflife* —The range is from 1 to 45 minutes. The default is 15. • *reuse* —The range is from is 1 to 20000 seconds. The default is 750. • *suppress* —The range is from is 1 to 20000. The default is 2000. • *duration* —The range is from is 1 to 255 minutes. The default is 60.
set extcomm-list *name* **delete** **Example:** `switch(config-route-map)# set extcomm-list BGPextCommunity delete`	Removes communities from the extended community attribute of an inbound or outbound BGP route update. Use the **ip extcommunity-list** command to create the extended community list.
set extcommunity generic { **transitive** | **nontransitive** }{ **none** | **additive**] *community-1* [*community-2...*]} **Example:** `switch(config-route-map)# set extcommunity generic transitive 1.0:30`	Sets the extended community attribute for a BGP route update. **Note** When you use both the **set extcommunity** and **set extcomm-list delete** commands in the same sequence of a route map attribute, the deletion operation is performed before the set operation. **Note** Use the **send-community** command in BGP neighbor address family configuration mode to propagate BGP extended community attributes to BGP peers.

Command	Purpose
set forwarding-address **Example:** `switch(config-route-map)# set forwarding-address`	Sets the forwarding address for OSPF.
set level { **backbone** | **level-1** | **level-1-2** | **level-2** } **Example:** `switch(config-route-map)# set level backbone`	Sets what area to import routes to for IS-IS. The options for IS-IS are level-1, level-1-2, or level-2. The default is level-1.
set local-preference *value* **Example:** `switch(config-route-map)# set local-preference 4000`	Sets the BGP local preference value. The range is from 0 to 4294967295.
set metric [+ | -] *bandwidth-metric* **Example :** `switch(config-route-map)# set metric +100`	Adds or subtracts from the existing metric value. The metric is in Kb/s. The range is from 0 to 4294967295.
set metric *bandwidth* [*delay reliability load mtu*] **Example :** `switch(config-route-map)# set metric 33 44 100 200 1500`	Sets the route metric values. Metrics are as follows: • *metric0* —Bandwidth in Kb/s. The range is from 0 to 4294967295. • *metric1* —Delay in 10-microsecond units. • *metric2* —Reliability. The range is from 0 to 255 (100 percent reliable). • *metric3* —Loading. The range is from 1 to 200 (100 percent loaded). • *metric4* —MTU of the path. The range is from 1 to 4294967295.
set metric-type { **external** | **internal** | **type-1** | **type-2** } **Example:** `switch(config-route-map)# set metric-type internal`	Sets the metric type for the destination routing protocol. The options are as follows: external—IS-IS external metric internal— IGP metric as the MED for BGP type-1—OSPF external type 1 metric type-2—OSPF external type 2 metric

Command	Purpose
set origin { **egp** *as-number* | **igp** | **incomplete**} **Example:** `switch(config-route-map)# set origin incomplete`	Sets the BGP origin attribute. The EGP *as-number* range is from 0 to 65535.
set tag *name* **Example:** `switch(config-route-map)# set tag 33`	Sets the tag value for the destination routing protocol. The *name* parameter is an unsigned integer.
set weight *count* **Example:** `switch(config-route-map)# set weight 33`	Sets the weight for the BGP route. The range is from 0 to 65535.

The **set metric-type internal** command affects an outgoing policy and an eBGP neighbor only. If you configure both the **metric** and **metric-type internal** commands in the same BGP peer outgoing policy, then Cisco NX-OS ignores the **metric-type internal** command.

Verifying the Route Policy Manager Configuration

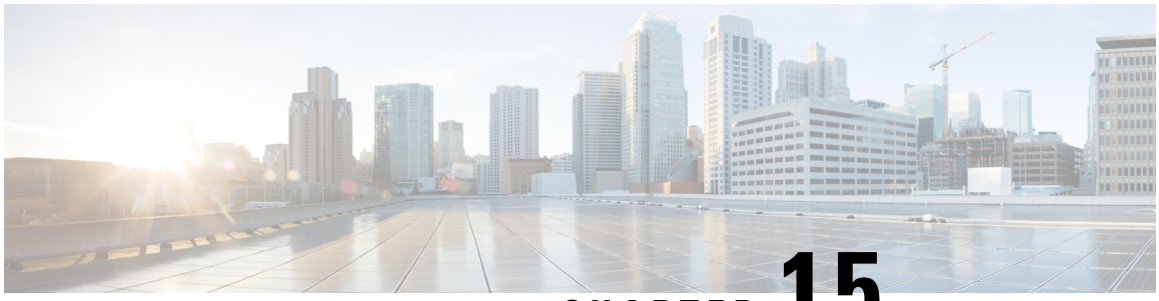
To display the route policy manager configuration information, perform one of the following tasks:

Command	Purpose
show ip community-list [*name*]	Displays information about a community list.
show ip ext community-list [*name*]	Displays information about an extended community list.
show [ip] prefix-list [*name*]	Displays information about an IPv4 prefix list.
show route-map [*name*]	Displays information about a route map.

Related Topics

The following topics can give more information on Route Policy Manager:

- Configuring Basic BGP



CHAPTER 15

Configuring Bidirectional Forwarding Detection

This chapter describes how to configure Bidirectional Forwarding Detection (BFD).

This chapter includes the following sections:

- Information About BFD, on page 309
- Prerequisites for BFD, on page 311
- Guidelines and Limitations, on page 311
- Default Settings, on page 312
- Configuring BFD, on page 313
- Configuring BFD for IPv6, on page 322
- Verifying the BFD Configuration, on page 327
- Monitoring BFD, on page 328

Information About BFD

BFD is a detection protocol designed to provide fast forwarding-path failure detection times for media types, encapsulations, topologies, and routing protocols. You can use BFD to detect forwarding path failures at a uniform rate, rather than the variable rates for different protocol hello mechanisms. BFD makes network profiling and planning easier and reconvergence time consistent and predictable.

BFD provides subsecond failure detection between two adjacent devices.

Asynchronous Mode

Cisco NX-OS supports the BFD asynchronous mode, which sends BFD control packets between two adjacent devices to activate and maintain BFD neighbor sessions between the devices. You configure BFD on both devices (or BFD neighbors). Once BFD has been enabled on the appropriate protocols, Cisco NX-OS creates a BFD session, negotiates BFD session parameters, and begins to send BFD control packets to each BFD neighbor at the negotiated interval. The BFD session parameters include the following:

- Desired minimum transmit interval—The interval at which this device wants to send BFD hello messages.
- Required minimum receive interval—The minimum interval at which this device can accept BFD hello messages from another BFD device.
- Detect multiplier—The number of missing BFD hello messages from another BFD device before this local device detects a fault in the forwarding path.

BFD Detection of Failures

Once a BFD session has been established and timer negotiations are complete, BFD neighbors send BFD control packets that act in the same manner as an IGP hello protocol to detect liveliness, except at a more accelerated rate. BFD detects a failure, but the protocol must take action to bypass a failed peer.

BFD sends a failure detection notice to the BFD-enabled protocols when it detects a failure in the forwarding path. The local device can then initiate the protocol recalculation process and reduce the overall network convergence time.

When a failure occurs in the network, the following occurs:

1. 1. The BFD neighbor session with the BFD neighbor router is torn down.
2. 2. BFD notifies the local BFD process that the BFD neighbor is no longer reachable.
3. 3. The local BFD process tears down the BFD neighbor relationship.
4. 4. If an alternative path is available, the routers immediately start converging on it.

Note The BFD failure detection occurs in less than a second.

BFD Echo Function

The BFD echo function sends echo packets from the forwarding engine to the remote BFD neighbor. The BFD neighbor forwards the echo packet back along the same path in order to perform detection; the BFD neighbor does not participate in the actual forwarding of the echo packets. The echo function and the forwarding engine are responsible for the detection process. BFD can use the slow timer to slow down the asynchronous session when the echo function is enabled and reduce the number of BFD control packets that are sent between two BFD neighbors. Also, the forwarding engine tests the forwarding path on the remote (neighbor) system without involving the remote system, so there is less interpacket delay variability and faster failure detection times.

The echo function is asymmetrical when both BFD neighbors are running echo function.

Security

Cisco NX-OS uses the packet Time to Live (TTL) value to verify that the BFD packets came from an adjacent BFD peer. For all asynchronous and echo request packets, the BFD neighbor sets the TTL value to 255 and the local BFD process verifies the TTL value as 255 before processing the incoming packet. For the echo response packet, BFD sets the TTL value to 254.

Virtualization Support

BFD supports virtual routing and forwarding instances (VRFs).

Prerequisites for BFD

BFD has the following prerequisites:

- You must enable the BFD feature (see the Enabling the BFD FeatureFeature section).
- For any client protocols that you want to enable BFD on, you enable BFD in that client protocol. See the Configuring BFD for IPv6 section.
- Disable Internet Control Message Protocol (ICMP) redirect messages on a BFD-enabled interfaces.
- See other detailed prerequisites that are listed with the configuration tasks.
- BFD is supported for BGP and PIM.

Guidelines and Limitations

BFD has the following configuration guidelines and limitations:

- BFD supports BFD version 1.
- BFD supports IPv4, IPv6, OSPFv2, BGPv4 and static routes.
- BFD supports single-hop BFD.
- BFD for BGP supports single-hop eBGP and iBGP with an update source.
- BFD supports the following Layer 3 interfaces—physical interfaces, port channels, subinterfaces, and VLAN interfaces (SVI).
- BFD supports authentication for all interfaces.
- BFD depends on a Layer 3 adjacency information to discover topology changes, including Layer 2 topology changes. A BFD session on a VLAN interface (SVI) may not be up after the convergence of the Layer 2 topology if there is no Layer 3 adjacency information available.
- Port channel configuration limitations:

– For Layer 3 port channels used by BFD, you must enable the Link Aggregation Control Protocol (LACP) on the port channel.

– For Layer 2 port channels used by SVI sessions, you must enable LACP on the port channel.

- SVI limitations:

– When you change the topology (for example, add or delete a link into a VLAN, delete a member from a Layer 2 port channel, and so on), the SVI session could be affected. It may go down first and then come up after the topology discovery is finished.

Tip If you do not want the SVI sessions to flap and you need to change the topology, you can disable the BFD feature before making the changes and reenable BFD after the changes have been made. You can also configure the BFD timer to be a large value (for example, 5 seconds), and change it back to a fast timer after the above events complete.

- Cisco NX-OS does not distribute the BFD operation to compatible modules to offload the CPU for BFD packet processing.
- BFD does not support stateless restarts and in-service software upgrades (ISSUs).
- We recommend that you configure LACP on the port channel if you want to enable BFD for a peer reachable through a port channel.
- BFD echo mode and Unicast Reverse Path Forwarding (URPF) are mutually exclusive and cannot both be enabled on a BFD interface. If you want to configure an interface for BFD, you must disable either BFD echo mode or URPF.
- Static IPv6 routes are supported with BFD.
- BFD does not support IPv6 echo mode. BFD uses IPv6 link local addresses only.

Default Settings

Table below lists the default settings for BFD parameters.

Table 21: Default BFD Parameters

Parameters	Default
BFD feature	Disabled
Required minimum receive interval	250 milliseconds
Desired minimum transmit interval	250 milliseconds
Echo Rx interval for the BFD session	50 milliseconds
Detect multiplier	3
Echo function	Enabled
Mode	Asynchronous
Port channel	Logical mode (one session per source-destination pair address)
Slow timer	2000 milliseconds
Startup timer	5 seconds

Configuring BFD

This section includes the following topics:

Configuration Hierarchy

You can configure BFD at the global level, VRF level, at the interface or port channel level, or at the subinterface level (for physical interfaces and port channels). The VRF configuration overrides global configuration. The interface or port channel configuration overrides VRF or global configuration. On supported interfaces, the subinterface-level configuration overrides the interface or port channel configuration unless subinterface optimization is enabled. See the Optimizing BFD on Subinterfaces section for more information.

For physical ports that are members of a port channel, the member port inherits the primary port channel BFD configuration. The member port subinterfaces can override the primary port channel BFD configuration, unless subinterface optimization is enabled.

Task Flow for Configuring BFD

Follow these steps to configure BFD:

Procedure

Step 1 Enabling the BFD Feature.

Step 2 Configuring Global BFD Parameters or Configuring BFD on an Interface.

Enabling the BFD Feature

You must enable the BFD feature before you can configure BFD on an interface and protocol.

Procedure

	Command or Action	Purpose

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

Step 2	**feature bfd** **Example:** `switch(config)# feature bfd`	Enables the BFD feature.
Step 3	(Optional) **copy running-config startup-config** **Example:**	Saves this configuration change.

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

Configuring Global BFD Parameters

You can configure the BFD session parameters for all BFD sessions on the device. The BFD session parameters are negotiated between the BFD peers in a three-way handshake.

See the Configuring BFD on an Interface section to override these global session parameters on an interface.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Procedure

	Command or Action	Purpose

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

Step 2	**bfd interval** *mintx* **min_rx** *msec* **multiplier** *value* **Example:** `switch(config)# bfd interval 250 min_rx 250 multiplier 3`	Configures the BFD session parameters for all BFD sessions on the device. You can override these values by configuring the BFD session parameters on an interface. The *mintx* and *msec* range is from 250 to 999 milliseconds and the default is 250. The multiplier range is from 3 to 50. The multiplier default is 3. To return to the default settings, use the **no bfd interval** command.
Step 3	**bfd slow-timer** [*interval*] **Example:** `switch(config)# bfd slow-timer 2000`	Configures the slow timer. This value determines how fast BFD starts up a new session and is used to slow down the asynchrounous sessions when the BFD echo function is enabled. The range is from 1000 to 30000 milliseconds. The default is 2000. To return to the default settings, use the no bfd slow-timer command.

Step 4	**exit**
	Example:
	`switch(config)# exit`
	`switch#`
		Returns to EXEC mode.

Step 5	(Optional) **show running-config bfd** **Example:**	Displays the BFD running configuration.

	Command or Action	Purpose
	`switch# show running-config bfd`	
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	Saves this configuration change.

Configuring BFD on an Interface

Enable the BFD feature. See the Enabling the BFD Feature section.

Before you begin

You can configure the BFD session parameters for all BFD sessions on an interface. The BFD session parameters are negotiated between the BFD peers in a three-way handshake.

This configuration overrides the global session parameters for the configured interface.

Procedure

	Command or Action	Purpose

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

Step 2	**interface** *int-if*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.

Step 3	**no ip redirect** **Example:** `switch(config-if)# no ip redirect`	Disables Internet Control Message Protocol (ICMP) redirect messages.
Step 4	**bfd interval** *mintx* **min_rx** *msec* **multiplier** *value* **Example:** `switch(config-if)# bfd interval 250 min_rx 250 multiplier 3`	Configures the BFD session parameters for all BFD sessions on the interface. This command overrides the global BFD session parameters. The *mintx* and *msec* range is from 250 to 999 milliseconds and the default is 250. The multiplier range is from 3 to 50. The multiplier default is 3. To return to the default settings, use the **no bfd interval** command.
Step 5	**exit** **Example:**	Exits interface configuration mode.

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

Step 6	**exit**
	Example:
	`switch (config)# exit`
	`switch#`
		Exits configuration mode and returns to EXEC mode.

Step 7	(Optional) **show running-config bfd** **Example:** `switch# show running-config bfd`	Displays the BFD running configuration.
Step 8	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	Saves this configuration change.

Configuring BFD on a Port Channel

You can configure the BFD session parameters for all BFD sessions on a port channel. For example, if the BFD session for one link on a port channel is up, BFD informs client protocols, such as BGP, that the port channel is up. The BFD session parameters are negotiated between the BFD peers in a three-way handshake.

This configuration overrides the global session parameters for the configured port channel. The member ports of the port channel inherit the port channel BFD session parameters, unless you configure subinterface-level BFD parameters on a member port. In that case, the member port subinterface uses the subinterface BFD configuration if subinterface optimization is not enabled. See the Optimizing BFD on Subinterfaces section for more information.

Before you begin

Ensure that you enable LACP on the port channel before you enable BFD.

Enable the BFD feature. See the Enabling the BFD Feature section.

Procedure

	Command or Action	Purpose

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

Step 2	**interface port-channel** *number*
	Example:
	`switch(config)# interface port-channel 2`
	`switch(config-if)#`
		Enters port channel configuration mode. Use the **?** keyword to display the supported number range.

	Command or Action	Purpose
Step 3	(Optional) **bfd interval** *mintx* **min_rx** *msec* **multiplier** *value* **Example:** `switch(config-if)# bfd interval 250 min_rx 250 multiplier 3`	Configures the BFD session parameters for all BFD sessions on the interface. This command overrides the global BFD session parameters. The *mintx* and *msec* range is from 250 to 999 milliseconds and the default is 250. The multiplier range is from 3 to 50. The multiplier default is 3. To return to the default settings, use the **no bfd interval** command.

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

Step 5	**exit**
	Example:
	`switch (config)# exit`
	`switch#`
		Exits configuration mode and returns to EXEC mode.

Step 6	(Optional) **show running-config bfd** **Example:** `switch# show running-config bfd`	Displays the BFD running configuration.
Step 7	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	Saves this configuration change.

Configuring the BFD Echo Function

You can configure the BFD echo function on one or both ends of a BFD-monitored link. The echo function slows down the required minimum receive interval, based on the configured slow timer. The RequiredMinEchoRx BFD session parameter is set to zero if the echo function is disabled. The slow timer becomes the required minimum receive interval if the echo function is enabled.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or theConfiguring BFD on an Interface section.

Ensure that Internet Control Message Protocol (ICMP) redirect messages are disabled on BFD-enabled interfaces. Use the **no ip redirects** command on the interface.

Procedure

	Command or Action	Purpose

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

Step 2	**bfd slow-timer** *echo-interval* **Example:** `switch(config)# bfd slow-timer 2000`	Configures the slow timer used in the echo function. This value determines how fast BFD starts up a new session and is used to slow down the asynchrounous sessions when the BFD echo function is enabled. This value overwrites the required minimum receive interval when the echo function is enabled. The range is from 1000 to 30000 milliseconds. The default is 2000. To return to the default settings, use the no bfd slow-timer command.

Step 3	**interface** *int-if*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.

Step 4	**bfd echo** **Example:** `switch(config-if)# bfd echo`	Enables the echo function. The default is enabled. To disable the echo function, use the no bfd echo command.

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

Step 6	**exit**
	Example:
	`switch (config)# exit`
	`switch#`
		Exits configuration mode and returns to EXEC mode.

Step 7	(Optional) **show running-config bfd** **Example:** `switch# show running-config bfd`	Displays the BFD running configuration.

Step 8	(Optional) **copy running-config startup-config**
	Example:
	`switch# copy running-config`
	`startup-config`
		Saves this configuration change.

Configuring BFD on BGP

You can configure BFD for the Border Gateway Protocol (BGP).

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *as-number*
	Example:
	`switch(config)# router bgp 64496`
	`switch(config-router)#`
		Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.

Step 3	**neighbor** { *ip-address* } **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 209.165.201.1 remote-as 64497`
	`switch(config-router-neighbor)#`
		Configures the IPv4 and AS number for a remote BGP peer. The *ip-address* format is x.x.x.x.

Step 4	**bfd** **Example:** `switch(config-router-neighbor)# bfd`	Enables BFD for this BGP peer.
Step 5	(Optional) **show running-config bfd** **Example:** `switch# show running-config bfd`	Displays the BFD running configuration.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	Saves this configuration change.

Configuring BFD on PIM

You can configure BFD for the Protocol Independent Multicast (PIM) protocol.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Enable the PIM feature. See the Cisco Nexus 3600 Switch NX-OS Multicast Routing Configuration Guide for more information.

Procedure

	Command or Action	Purpose

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

Step 2	**ip pim bfd** **Example:** `switch(config)# ip pim bfd`	Enables BFD for PIM.

Step 3	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode. Use the **?** keyword to display the supported interfaces.

Step 4	(Optional) **ip pim bfd-instance** [**disable**] **Example:** `switch(config-if)# ip pim bfd-instance`	Enables or disables BFD on a PIM interface. The default is disabled.
Step 5	(Optional) **show running-config pim** **Example:** `switch(config)# show running-config pim`	Displays the PIM running configuration.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	Saves this configuration change.

Configuring BFD on OSPFv2

You can configure BFD for the Open Shortest Path First Protocol (OSPFv2).

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Configure the BFD session parameters. See the Configuring Global BFD Parameters section or the Configuring BFD on an Interface section.

Enable the OSPFv2 feature.

Procedure

	Command or Action	Purpose

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

Step 2	**router ospf** *process-id*
	Example:
	`switch(config)# router ospf 64496`
	`switch(config-router)#`
		Creates a new OSPFv2 process with a configured id.

Step 3	**bfd** **Example:** `switch(config-router)# bfd`	Enables BFD for this OSPFv2 peer. The default value is disabled.
Step 4	(Optional) **show running-config ospf** **Example:** `switch(config)# show running-config ospf`	Displays the OSPFv2 running configuration.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	Saves this configuration change.

Configuring BFD for Static Routes

You can configure BFD for static routes on an interface. You can optionally configure BFD on a static route within a virtual routing and forwarding (VRF) instance.

Before you begin

Enable the BFD feature. See the Enabling the BFD Feature section.

Procedure

	Command or Action	Purpose

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

Step 2	(Optional) **vrf context** *vrf-name*
	Example:
	`switch(config)# vrf context Red`
	`switch(config-vrf)#`
		Enters VRF configuration mode.

	Command or Action	Purpose
Step 3	**ip route** *route interface* { *nh-address* | *nh-prefix* } **Example:** `switch(config-vrf)# ip route 192.0.2.1 ethernet 2/1 192.0.2.4`	Creates a static route Use the **?** keyword to display the supported interfaces.
Step 4	**ip route static bfd** *interface* { *nh-address* | *nh-prefix* } **Example:** `switch(config-vrf)# ip route static bfd ethernet 2/1 192.0.2.4`	Enables BFD for all static routes on an interface. Use the **?** keyword to display the supported interfaces.
Step 5	(Optional) **show ip route static** [**vrf** *vrf-name*] **Example:** `switch(config-vrf)# show ip route static vrf Red`	Displays the static routes.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	Saves this configuration change.

Configuring BFD for IPv6

Configuring Global BFD Parameters for IPv6

1. 1. configure terminal
2. 2. bfd [ipv4 | ipv6] interval [interval min_rx milliseconds multiplier interval-multiplier]

Before you begin

You can specify either the IPv4 or the IPv6 address family when you configure BFD parameters.

Procedure

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

	Command or Action	Purpose
Step 2	**bfd [ipv4 | ipv6] interval [interval min_rx milliseconds multiplier interval-multiplier]** **Example:** `switch(config)# bfd ipv6 interval 50 min_rx 50 multiplier 3`	Configures the BFD session parameters for all BFD sessions in the specified address family on the device. The Tx and Rx intervals range between 50 and 999 milliseconds. The multiplier ranges between 3 and 50.

Configuring Per Interface BFD Parameters for IPv6

Before you begin

BFD must be enabled on the device.

Procedure

	Command or Action	Purpose

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

Step 2	**interface** *interface*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Use the **?** keyword to display the supported interfaces

Step 3	**bfd [ipv4 | ipv6] interval [interval** *min_rx milliseconds* **multiplier** *interval-multiplier* **Example:** `switch(config-if)# bfd ipv6 interval 50 min_rx 50 multiplier 3`	Configures the BFD session parameters for all BFD sessions in the specified address family on the device. The Tx and Rx intervals range between 50 and 999 milliseconds. The multiplier ranges between 3 and 50.

Configuring BFD for IPv6 on OSPFv3

You can configure BFD for IPv6 on the Open Shortest Path First Protocol (OSPFv3).

Before you begin

- Enable the BFD feature. See the Enabling the BFD Feature section.
- Enable the OSPFv3 feature.
- Configure the BFD session parameters. See the Configuring Global BFD Parameters for IPv6 section or the Configuring Per Interface BFD Parameters for IPv6 section.

- Enable the OSPFv3 feature.

Procedure

	Command or Action	Purpose

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

Step 2	**router ospfv3** *process-id*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv2 process with a configured id.

Step 3	**bfd** **Example:** `switch(config-router)# bfd`	Enables BFD for this OSPFv3 peer. The default value is disabled.
Step 4	(Optional) **show running-config ospfv3** **Example:** `switch(config-router)# show running-config ospfv3`	Displays the OSPFv3 running configuration.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	Saves this configuration change.

Configuring BFD on IPv6 Static Routes

You can configure BFD for all IPv6 static routes on an interface.

Before you begin

- Ensure that BFD is enabled on the devices at each end of the static route.
- Ensure that the BFD session parameters are configured.

Procedure

	Command or Action	Purpose

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

	Command or Action	Purpose

Step 2	(Optional) **vrf context** *vrf-name*
	Example:
	`switch(config)# vrf context Red`
	`switch(config-vrf)#`
		Enters VRF configuration mode to configure BFD on an IPv6 static route.
		Specifies the VRF for the route to be tracked.

Step 3	**ipv6 route** *route interface* { *nh-address* | *nh-prefix* } **Example:** `switch(config-vrf)# ipv6 route 1::5/64 ethernet 1/3 2::2`	Creates an IPv6 static route. • Specify the IPv6 address for the route argument • Use the **?** keyword to display the supported interfaces. • Specify the next-hop (nh) address or prefix for this static route.
Step 4	**ipv6 route static bfd** *network-interface* {*nh-address* | *nh-prefix* } **Example:** `switch(config-vrf)# ipv6 route static bfd ethernet 1/3 2::2`	Enables BFD for all IPv6 static routes on an interface. • Use the **?** keyword to display the supported interfaces. • Specify the next-hop (nh) address or prefix for this static route.
Step 5	(Optional) **show bfd neighbors** **Example:** `switch(config-vrf)# show bfd neighbors`	Displays information about BFD neighbors.
Step 6	(Optional) **show ipv6 route static** **Example:** `switch(config-vrf)# show ipv6 route static vrf Red`	Displays the static routes.
Step 7	(Optional) **copy running-config startup-config** **Example:** `switch(config-vrf)# copy running-config startup-config`	Saves this configuration change.

Configuring BFD Echo Mode

The echo function is enabled by default. You can disable it if desired.

Procedure

	Command or Action	Purpose

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

Step 2	**interface type slot/port**
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Use the **?** keyword to display the supported interfaces

Step 3	**[no] bfd ipv4 echo** **Example:** `switch(config-if)# bfd ipv4 echo`	Enables the echo function. The default is enabled. To disable the echo function for the specified address family, use the no form of the command.

Configuring BFD Session Echo Interval

You can configure the echo Rx interval for BFD sessions.

Procedure

	Command or Action	Purpose

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

Step 2	**interface** *interface*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Use the **?** keyword to display the supported interfaces

Step 3	**[no] bfd ipv4 echo-rx-interval interval** **Example:** `switch(config-if)# bfd ipv4 echo-rx-interval 500`	Configures the echo Rx interval for the BFD session. The interval can range between 50 and 999 milliseconds. To return the echo interval to the default value of 50 milliseconds, use the no form of the command.

Configuring a BFD Echo Interface

Perform this task to configure the loopback interface as the source address for echo frames.

Procedure

	Command or Action	Purpose

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

Step 2	**interface loopback number**
	Example:
	`switch(config)# interface loopback 50`
	`switch(config-if)#`
		Creates a loopback interface and enters interface configuration mode.

Step 3	**ip address ip-address mask** **Example:** `switch(config-if)# ip address 192.108.1.27 255.255.255.0`	Configures the IP address as the source address for echo frames.

Configuring the BFD Slow Timer

Echo mode is enabled by default. You can configure the slow-timer value and disable or enable echo mode for an address family.

Procedure

	Command or Action	Purpose

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

Step 2	**interface** *interface*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Use the **?** keyword to display the supported interfaces

Step 3	**bfd ipv4 slow-timer [interval]** **Example:** `switch(config-if)# bfd ipv4 slow-timer 6000`	Configures the slow timer, in milliseconds, used in the echo function for the specified address family.

Verifying the BFD Configuration

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

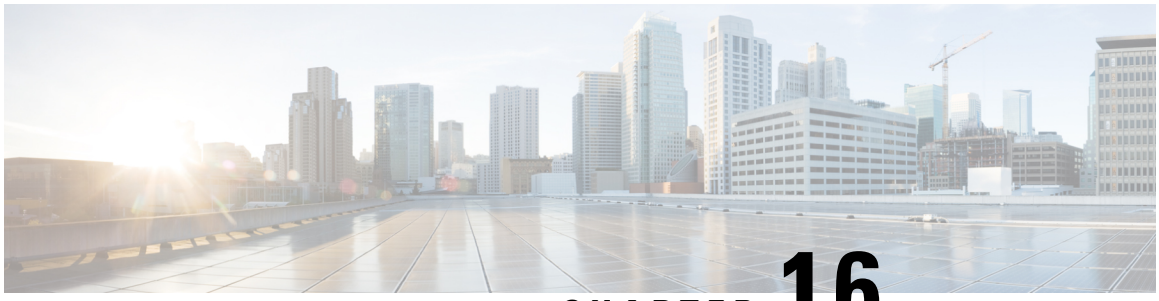
Command	Purpose
show running-config bfd	Displays the running BFD configuration.
show startup-config bfd	Displays the BFD configuration that will be applied on the next system startup.

Monitoring BFD

Use the following commands to display BFD:

Command	Purpose
show bfd neighbors [**application** *name*] [**details**]	Displays information about BFD for a supported application, such as BGP.
show bfd neighbors [**interface** *int-if*] [**details**]	Displays information about BGP sessions on an interface.
show bfd neighbors [**dest-ip** *ip-address*] [**src-ip** *ip-address*][*details*]	Displays information about the specified BGP session on an interface.
show bfd neighbors [**vrf** *vrf-name*] [**details**]	Displays information about BFD for a VRF.

For detailed information about the fields in the output from these commands, see the Cisco Nexus 3548 Switch Command Reference.



CHAPTER 16

Configuring Policy-Based Routing

This chapter describes how to configure policy based routing on the Cisco NX-OS device.

This chapter includes the following sections:

- About Policy-Based Routing, on page 329
- Prerequisites for Policy-Based Routing, on page 331
- Guidelines and Limitations for Policy-Based Routing, on page 331
- Default Settings, on page 332
- Configuring Policy-Based Routing, on page 332
- Verifying the Policy-Based Routing Configuration, on page 334
- Configuration Examples for Policy Based-Routing, on page 334

About Policy-Based Routing

With policy-based routing, you can configure a defined policy for IPv4 and IPv6 traffic flows that lessens the reliance on routes derived from routing protocols. All packets received on an interface with policy-based routing enabled are passed through enhanced packet filters or route maps. The route maps dictate the policy that determines where to forward packets.

Policy-based routing includes the following features:

- Source-based routing—Routes traffic that originates from different sets of users through different connections across the policy routers.
- Quality of Service (QoS)—Differentiates traffic by setting the precedence or type of service (ToS) values in the IP packet headers at the periphery of the network and leveraging queuing mechanisms to prioritize traffic in the core or backbone of the network (see the Cisco Nexus 3600 NX-OS Quality of Service Configuration Guide).
- Load sharing—Distributes traffic among multiple paths based on the traffic characteristics.

Policy Route Maps

Each entry in a route map contains a combination of match and set statements. The match statements define the criteria for whether appropriate packets meet the particular policy (that is, the conditions to be met). The set clauses explain how the packets should be routed once they have met the match criteria.

You can mark the route-map statements as permit or deny. You can interpret the statements as follows:

- If the statement is marked as permit and the packets meet the match criteria, the set clause is applied. One of these actions involves choosing the next hop.
- If a statement is marked as deny, the packets that meet the match criteria are sent back through the normal forwarding channels, and destination-based routing is performed.
- If the statement is marked as permit and the packets do not match any route-map statements, the packets are sent back through the normal forwarding channels, and destination-based routing is performed.

Note Policy routing is specified on the interface that receives the packets, not on the interface from which the packets are sent.

Set Criteria for Policy-Based Routing

The Cisco Nexus 3600 platform switches support the following set commands for route maps used in policy-based routing:

- set {ip | ipv6} next-hop address1 [address2...] [load-share]
- set interface null0

These set commands are mutually exclusive within the route-map sequence.

In the first command, the IP address specifies the adjacent next-hop router in the path toward the destination to which the packets should be forwarded. The first IP address associated with a currently up connected interface is used to route the packets.

Note You can optionally configure this command for next-hop addresses to load balance traffic for up to 32 IP addresses. In this case, Cisco NX-OS sends all traffic for each IP flow to a particular IP next-hop address.

If the packets do not meet any of the defined match criteria, those packets are routed through the normal destination-based routing process.

Route-Map Processing Logic

When a packet is received on an interface that is configured with a route map, the forwarding logic processes each route-map statement according to the sequence number.

If the route-map statement encountered is a route-map...permit statement, the packet is matched against the criteria in the match command. This command may refer to an ACL that has one or more access control entries (ACEs). If the packet matches the permit ACEs in the ACL, the policy-based routing logic executes the action specified by the set command on the packet.

If the route-map statement encountered is a route-map... deny statement, the packet is matched against the criteria in the match command. This command may refer to an ACL that has one or more ACEs. If the packet matches the permit ACEs in the ACL, policy-based routing processing terminates, and the packet is routed using the default IP routing table.

Note The set command has no effect inside a route-map... deny statement.

- If the route-map configuration does not contain a match statement, the policy-based routing logic executes the action specified by the set command on the packet. All packets are routed using policy-based routing.
- If the route-map configuration references a match statement but the match statement references a non-existing ACL or an existing ACL without any access control entries (ACEs), the packet is routed using the default routing table.
- If the next-hop specified in the set {ip | ipv6} next-hop command is down, is not reachable, or is removed, the packet is routed using the default routing table.

Policy-Based Routing Filtering Options

You can identify traffic by using additional options. The following list includes most but not all additional filtering options.

Policy-based routing ACLs support the following additional filtering options:

- Layer 3 source and/or destination address
- TCP and UDP ports

Prerequisites for Policy-Based Routing

You can identify traffic by using additional options. The following list includes most but not all additional filtering options.

Policy-based routing ACLs support the following additional filtering options:

- Layer 3 source and/or destination address
- TCP and UDP ports

Guidelines and Limitations for Policy-Based Routing

Policy-based routing has the following configuration guidelines and limitations:

- Beginning with Cisco NX-OS Release 7.0(3)F3(3), Cisco Nexus 3600 platform switches support IPv4 and IPv6 policy-based routing. For these switches, PBR policy has a higher priority over attached and local routes. Explicit allowed listing might be required if protocol neighbors are directly attached.
- A policy-based routing route map can have only one match statement per route-map statement.
- A match command cannot refer to more than one ACL in a route map used for policy-based routing.
- The same route map can be shared among different interfaces for policy-based routing as long as the interfaces belong to the same virtual routing and forwarding (VRF) instance.

- Using a prefix list as a match criteria is not supported. Do not use a prefix list in a policy-based routing route map.
- Policy-based routing supports only unicast traffic. Multicast traffic is not supported.
- Policy-based routing is supported with Layer 3 port-channel subinterfaces.
- An ACL used in a policy-based routing route map cannot include deny access control entries (ACEs).
- Policy-based routing is supported only in the default system routing mode.
- Policy-based routing traffic cannot be balanced if the next hop is recursive over ECMP paths. Instead, use the **set {ip | ipv6} next-hop** *ip-address* **load-share** command to specify the adjacent next hops.
- Policy-based routing is not supported with VXLAN.
- Policy-based routing policy statistics are not supported.

Default Settings

Table below lists the default settings for policy-based routing parameters.

Table 22: Default Policy-based Routing Parameters

Parameters	Default
Policy-based routing	Disabled

Configuring Policy-Based Routing

Enabling the Policy-Based Routing Feature

You must enable the policy-based routing feature before you can configure a route policy.

Procedure

	Command or Action	Purpose

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

Step 2	[**no**] **feature pbr** **Example:** `switch(config)# feature pbr`	Enables the policy-based routing feature. Use the no form of this command to disable the policy-based routing feature.

	Command or Action	Purpose
		Note The no feature pbr command removes the policies applied under the interfaces. It does not remove the ACL or route-map configuration nor does it create a system checkpoint.
Step 3	(Optional) **show feature** **Example:** `switch(config)# show feature`	Displays enabled and disabled features.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves this configuration change.

Configuring a Route Policy

You can use route maps in policy-based routing to assign routing policies to the inbound interface.

Procedure

	Command or Action	Purpose

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

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

Step 3	{ **ip** | **ipv6** } **policy route-map** *map-name* **Example:** `switch(config-if)# ip policy route-map Testmap`	Assigns a route map for IPv4 or IPv6 policy-based routing to the interface.

Step 4	**route-map map-name [permit | deny] [seq]**
	Example:
	`switch(config-if)# route-map Testmap`
	`switch(config-route-map)#`
		Creates a route map or enters route-map configuration mode for an existing route map. Use seq to order the entries in a route map.

Step 5	**match {ip | ipv6} address access-list-name name** [*name...*]	Matches an IPv4 or IPv6 address against one or more IP or IPv6 access control lists (ACLs).

	Command or Action	Purpose
	Example: `switch(config-route-map)# match ip address access-list-name ACL1`	This command is used for policy-based routing and is ignored by route filtering or redistribution.
Step 6	(Optional) **set ip next-hop address1 [address2...]** [*load-share*] **Example:** `switch(config-route-map)# set ip next-hop 192.0.2.1`	Sets the IPv4 next-hop address for policy-based routing. This command uses the first valid next-hop address if multiple addresses are configured. Use the optional load-share keyword to load balance traffic across a maximum of 32 next-hop addresses.
Step 7	**set ipv6 next-hop address1 [address2...]** [*load-share*] **Example:** `switch(config-route-map)# set ipv6 next-hop 2001:0DB8::1`	Sets the IPv6 next-hop address for policy-based routing. This command uses the first valid next-hop address if multiple addresses are configured. Use the optional load-share keyword to load balance traffic across a maximum of 32 next-hop addresses.
Step 8	(Optional) **set interface null0** **Example:** `switch(config-route-map)# set interface null0`	Sets the interface used for routing. Use the null0 interface to drop packets.
Step 9	(Optional) **copy running-config startup-config** **Example:** `switch(config-route-map)# copy running-config startup-config`	Saves this configuration change.

Verifying the Policy-Based Routing Configuration

To display policy-based routing configuration information, perform one of the following tasks:

Command	Purpose
show [ip | ipv6] policy [*name*]	Displays information about an IPv4 or IPv6 policy.

Configuration Examples for Policy Based-Routing

This example shows how to configure a simple route policy on an interface:

```
ip access-list pbr-sample
permit tcp host 10.1.1.1 host 192.168.2.1 eq 80
!
route-map pbr-sample
```

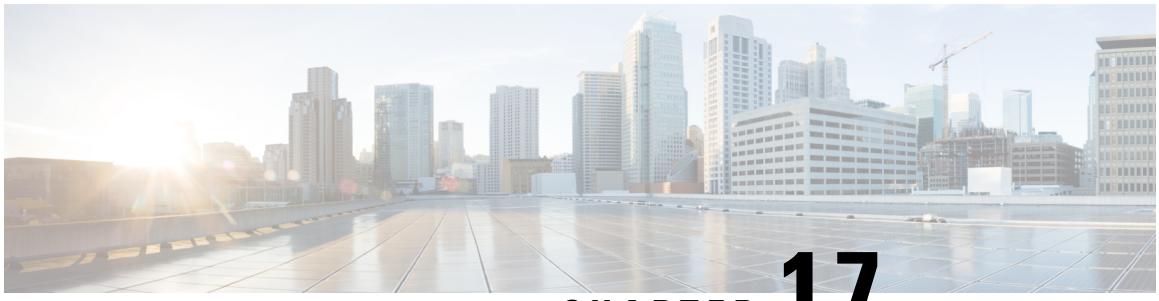
```
match ip address pbr-sampl
set ip next-hop 192.168.1.1
!
route-map pbr-sample
interface ethernet 1/2
ip policy route-map pbr-sample
```

The following output verifies this configuration:

```
switch# show route-map pbr-sample
route-map pbr-sample, permit, sequence 10
Match clauses:

ip address (access-lists): pbr-sample
Set clauses:

ip next-hop 192.168.1.1
switch# show ip policy
Interface Route-map Status VRF-Name
Ethernet1/2 pbr-sample Active --
```



CHAPTER 17

Configuring HSRP

This chapter describes how to configure the Hot Standby Router Protocol (HSRP) on Cisco NX-OS switches.

This chapter includes the following sections:

- Information About HSRP, on page 337
- Prerequisites for HSRP, on page 341
- Guidelines and Limitations for HSRP, on page 341
- Default Settings for HSRP, on page 342
- Configuring HSRP, on page 343
- Verifying the HSRP Configuration, on page 352
- Configuration Examples for HSRP, on page 352
- Additional References, on page 353

Information About HSRP

HSRP is a first-hop redundancy protocol (FHRP) that allows a transparent failover of the first-hop IP router. HSRP provides first-hop routing redundancy for IP hosts on Ethernet networks configured with a default router IP address. You use HSRP in a group of routers for selecting an active router and a standby router. In a group of routers, the active router is the router that routes packets; the standby router is the router that takes over when the active router fails or when preset conditions are met.

Many host implementations do not support any dynamic router discovery mechanisms but can be configured with a default router. Running a dynamic router discovery mechanism on every host is not feasible for a number of reasons, including administrative overhead, processing overhead, and security issues. HSRP provides failover services to these hosts.

HSRP Overview

When you use HSRP, you configure the HSRP virtual IP address as the host's default router (instead of the IP address of the actual router). The virtual IP address is an IPv4address that is shared among a group of routers that run HSRP.

When you configure HSRP on a network segment, you provide a virtual MAC address and a virtual IP address for the HSRP group. You configure the same virtual address on each HSRP-enabled interface in the group. You also configure a unique IP address and MAC address on each interface that acts as the real address. HSRP selects one of these interfaces to be the active router. The active router receives and routes packets destined for the virtual MAC address of the group.

HSRP detects when the designated active router fails. At that point, a selected standby router assumes control of the virtual MAC and IP addresses of the HSRP group. HSRP also selects a new standby router at that time.

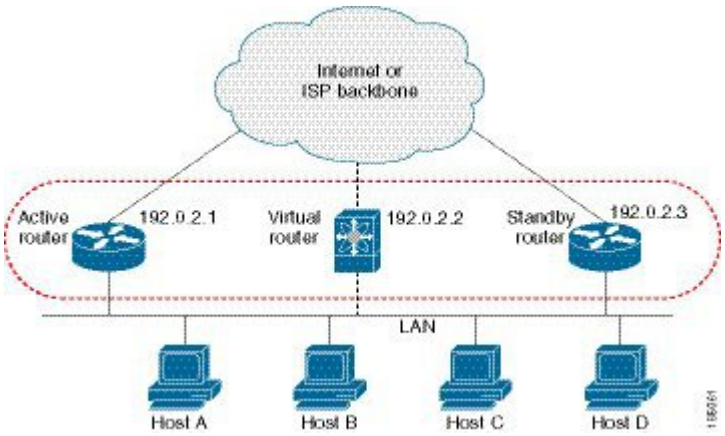
HSRP uses a priority mechanism to determine which HSRP-configured interface becomes the default active router. To configure an interface as the active router, you assign it with a priority that is higher than the priority of all the other HSRP-configured interfaces in the group. The default priority is 100, so if you configure just one interface with a higher priority, that interface becomes the default active router.

Interfaces that run HSRP send and receive multicast User Datagram Protocol (UDP)-based hello messages to detect a failure and to designate active and standby routers. When the active router fails to send a hello message within a configurable period of time, the standby router with the highest priority becomes the active router. The transition of packet forwarding functions between the active and standby router is completely transparent to all hosts on the network.

You can configure multiple HSRP groups on an interface.

The following figure shows a network configured for HSRP. By sharing a virtual MAC address and a virtual IP address, two or more interfaces can act as a single virtual router.

Figure 34: HSRP Topology with Two Enabled Routers



The virtual router does not physically exist but represents the common default router for interfaces that are configured to provide backup to each other. You do not need to configure the hosts on the LAN with the IP address of the active router. Instead, you configure them with the IP address (virtual IP address) of the virtual router as their default router. If the active router fails to send a hello message within the configurable period of time, the standby router takes over, responds to the virtual addresses, and becomes the active router, assuming the active router duties. From the host perspective, the virtual router remains the same.

Note Packets received on a routed port destined for the HSRP virtual IP address will terminate on the local router, regardless of whether that router is the active HSRP router or the standby HSRP router. This includes ping and Telnet traffic. Packets received on a Layer 2 (VLAN) interface destined for the HSRP virtual IP address will terminate on the active router.

HSRP for IPv4

HSRP routers communicate with each other by exchanging HSRP hello packets. These packets are sent to the destination IP multicast address 224.0.0.2 (reserved multicast address used to communicate to all routers)

on UDP port 1985. The active router sources hello packets from its configured IP address and the HSRP virtual MAC address while the standby router sources hellos from its configured IP address and the interface MAC address, which may or may not be the burned-in address (BIA). The BIA is the last six bytes of the MAC address that is assigned by the manufacturer of the network interface card (NIC).

Because hosts are configured with their default router as the HSRP virtual IP address, hosts must communicate with the MAC address associated with the HSRP virtual IP address. This MAC address is a virtual MAC address, 0000.0C07.ACxy, where xy is the HSRP group number in hexadecimal based on the respective interface. For example, HSRP group 1 uses the HSRP virtual MAC address of 0000.0C07.AC01. Hosts on the adjoining LAN segment use the normal Address Resolution Protocol (ARP) process to resolve the associated MAC addresses.

HSRP version 2 uses the new IP multicast address 224.0.0.102 to send hello packets instead of the multicast address of 224.0.0.2, which is used by version 1. HSRP version 2 permits an expanded group number range of 0 to 4095 and uses a new MAC address range of 0000.0C9F.F000 to 0000.0C9F.FFFF.

HSRP Versions

Cisco NX-OS supports HSRP version 1 by default. You can configure an interface to use HSRP version 2.

HSRP version 2 has the following enhancements to HSRP version 1:

- Expands the group number range. HSRP version 1 supports group numbers from 0 to 255. HSRP version 2 supports group numbers from 0 to 4095.
- For IPv4, uses the IPv4 multicast address 224.0.0.102to send hello packets instead of the multicast address of 224.0.0.2, which is used by HSRP version 1.
- Uses the MAC address range from 0000.0C9F.F000 to 0000.0C9F.FFFF. HSRP version 1 uses the MAC address range 0000.0C07.AC00 to 0000.0C07.ACFF.
- Adds support for MD5 authentication.

When you change the HSRP version, Cisco NX-OS reinitializes the group because it now has a new virtual MAC address.

HSRP version 2 has a different packet format than HSRP version 1. The packet format uses a type-length-value (TLV) format. HSRP version 2 packets received by an HSRP version 1 router are ignored.

HSRP Subnet VIP

Beginning with Cisco NX-OS Release 7.0(3)F3(3), you can configure an HSRP subnet virtual IP (VIP) address in a different subnet than that of the interface IP address.

This feature enables you to conserve public IPv4 addresses by using a VIP as a public IP address and an interface IP as a private IP address. HSRP subnet VIPs are not needed for IPv6 addresses because a larger pool of IPv6 addresses is available and because routable IPv6 addresses can be configured on an SVI and used with regular HSRP.

This feature also enables periodic ARP synchronization to vPC peers and allows ARP to source with the VIP when an HSRP subnet VIP is configured for hosts in the VIP subnet.

For more information, see Guidelines and Limitations for HSRP and Configuration Examples for HSRP.

HSRP Authentication

HSRP message digest 5 (MD5) algorithm authentication protects against HSRP-spoofing software and uses the industry-standard MD5 algorithm for improved reliability and security. HSRP includes the IPv4 address in the authentication TLVs.

HSRP Messages

Routers that are configured with HSRP exchange the following three types of multicast messages:

- Hello—The hello message conveys the HSRP priority and state information of the router to other HSRP routers.
- Coup—When a standby router wants to assume the function of the active router, it sends a coup message.
- Resign—The active router sends this message when it no longer wants to function as the active router.

HSRP Load Sharing

HSRP allows you to configure multiple groups on an interface. You can configure two overlapping IPv4 HSRP groups to load share traffic from the connected hosts while providing the default router redundancy expected from HSRP. Figure below shows an example of a load-sharing HSRP IPv4 configuration.

Figure 35: HSRP Load Sharing

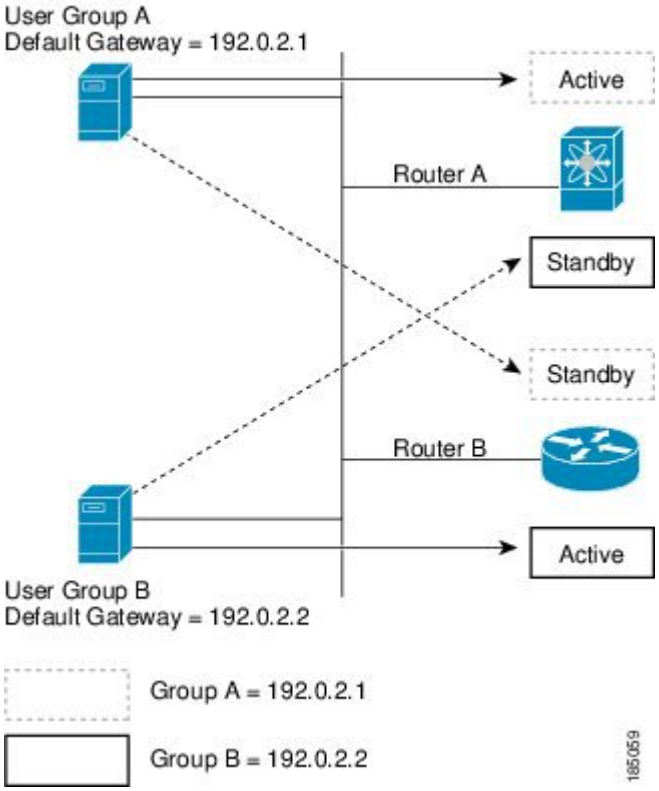


Figure **HSRP Load Sharing** shows two routers (A and B) and two HSRP groups. Router A is the active router for group A but is the standby router for group B. Similarly, router B is the active router for group B and the standby router for group A. If both routers remain active, HSRP load balances the traffic from the hosts across both routers. If either router fails, the remaining router continues to process traffic for both hosts.

Object Tracking and HSRP

You can use object tracking to modify the priority of an HSRP interface based on the operational state of another interface. Object tracking allows you to route to a standby router if the interface to the main network fails.

Two objects that you can track are the line protocol state of an interface or the reachability of an IP route. If the specified object goes down, Cisco NX-OS reduces the HSRP priority by the configured amount. For more information, see the Configuring HSRP Object Tracking section.

Virtualization Support

HSRP supports Virtual Routing and Forwarding instances (VRFs). By default, Cisco NX-OS places you in the default VRF unless you specifically configure another VRF.

If you change the VRF membership of an interface, Cisco NX-OS removes all Layer 3 configuration, including HSRP.

For more information, see Configuring Layer 3 Virtualization

Prerequisites for HSRP

HSRP has the following prerequisites:

- You must enable the HSRP feature in a switch before you can configure and enable any HSRP groups.

Guidelines and Limitations for HSRP

HSRP has the following configuration guidelines and limitations:

- The minimum hello timer value is 250 milliseconds.
- The minimum hold timer value is 750 milliseconds.
- You must configure an IP address for the interface that you configure HSRP on and enable that interface before HSRP becomes active.
- For IPv4, the virtual IP address must be in the same subnet as the interface IP address.
- We recommend that you do not configure more than one first-hop redundancy protocol on the same interface.
- HSRP version 2 does not interoperate with HSRP version 1. An interface cannot operate both version 1 and version 2 because both versions are mutually exclusive. However, the different versions can be run on different physical interfaces of the same router.

- You cannot change from version 2 to version 1 if you have configured groups above the group number range allowed for version 1 (0 to 255).
- Cisco NX-OS removes all Layer 3 configuration on an interface when you change the interface VRF membership, port channel membership, or when you change the port mode to Layer 2.
- The HSRP subnet VIP feature, which is introduced in Cisco NX-OS Release 7.0(3)F3(3), has the following guidelines and limitations:
  - – This feature is supported only for IPv4 addresses and only in a vPC topology.
  - – Primary or secondary VIPs can be subnet VIPs, but subnet VIPs must not overlap any interface subnet.
  - – Regular host VIPs use a mask length of 0 or 32. If you specify a mask length for a subnet VIP, it must be greater than 0 and less than 32.
  - – uRPF is not supported with this feature.
  - – DHCP sourcing with VIPs is also not supported.
  - – This feature does not support using a DHCP relay agent to relay DHCP packets with a VIP as the source.
  - – VIP direct routes must be explicitly advertised to routing protocols using redistribute commands and route maps.
  - – Supervisor-generated traffic (pings, trace routes, and so on) destined for VIP subnets will continue to source with SVI IP addresses and not with the VIP.
  - – If the subnet VIP is configured with /32 as the prefix, you must use the no command with the /32 prefix to remove the IP address (for example, no ip ip-address/32).

Default Settings for HSRP

Table below lists the default settings for HSRP parameters.

Table 23: Default HSRP Parameters

Parameters	Default
HSRP	Disabled
Authentication	Enabled as text for version 1, with cisco as the password
HSRP version	Version 1
Preemption	disabled
Priority	100
virtual MAC address	Derived from HSRP group number

Configuring HSRP

Enabling the HSRP Feature

You must globally enable the HSRP feature before you can configure and enable any HSRP groups.

DETAILED STEPS

To enable the HSRP feature, use the following command in global configuration mode:

Command	Purpose
feature hsrp **Example** : `switch(config)# feature hsrp`	Enables HSRP.

To disable the HSRP feature and remove all associated configuration, use the following command in global configuration mode:

Command	Purpose
no feature hsrp **Example** : `switch(config)# no feature hsrp`	Disables HSRP.

Configuring the HSRP Version

You can configure the HSRP version. If you change the version for existing groups, Cisco NX-OS reinitializes HSRP for those groups because the virtual MAC address changes. The HSRP version applies to all groups on the interface.

To configure the HSRP version, use the following command in interface configuration mode:

Command	Purpose
hsrp version { **1** | **2** } **Example** : `switch(config-if)# hsrp version 2`	Configures the HSRP version. Version 1 is the default.

Configuring an HSRP Group for IPv4

You can configure an HSRP group on an IPv4 interface and configure the virtual IP address and virtual MAC address for the HSRP group.

Before you begin

Ensure that you have enabled the HSRP feature (see the Enabling the HSRP Feature section).

Cisco NX-OS enables an HSRP group once you configure the virtual IP address on any member interface in the group. You should configure HSRP attributes such as authentication, timers, and priority before you enable the HSRP group.

Procedure

	Command or Action	Purpose

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

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

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 4	**ip address** *ip-address/length* **Example:** `switch(config-if)# ip address 192.0.2.2/8`	Configures the IPv4 address of the interface.

Step 5	**hsrp** *group-number* [**ipv4**]
	Example:
	`switch(config-if)# hsrp 2`
	`switch(config-if-hsrp)`
		Creates an HSRP group and enters hsrp configuration mode. The range for HSRP version 1 is from 0 to 255. The range is for HSRP version 2 is from 0 to 4095. The default value is 0.

Step 6	**ip** [*ip-address* [**secondary**]] **Example:** `switch(config-if-hsrp)# ip 192.0.2.1`	Configures the virtual IP address for the HSRP group and enables the group. This address should be in the same subnet as the IPv4 address of the interface.
Step 7	**exit** **Example:** `switch(config-if-hsrp)# exit`	Exits HSRP configuration mode.
Step 8	**no shutdown** **Example:** `switch(config-if)# no shutdown`	Enables the interface.

	Command or Action	Purpose
Step 9	(Optional) **show hsrp** [**group** *group-number*] [**ipv4**] **Example:** `switch(config-if)# show hsrp group 2`	Displays HSRP information.
Step 10	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

Note You should use the **no shutdown** command to enable the interface after you finish the configuration.

This example shows how to configure an HSRP group on Ethernet 1/2:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# ip 192.0.2.2/8
switch(config-if)# hsrp 2
switch(config-if-hsrp)# ip 192.0.2.1
switch(config-if-hsrp)# exit
switch(config-if)# no shutdown
switch(config-if)# copy running-config startup-config
```

Configuring the HSRP Virtual MAC Address

You can override the default virtual MAC address that HSRP derives from the configured group number.

To manually configure the virtual MAC address for an HSRP group, use the following command in hsrp configuration mode:

Command	Purpose
mac-address *string* **Example** : `switch(config-if-hsrp)# mac-address 5000.1000.1060`	Configures the virtual MAC address for an HSRP group. The string uses the standard MAC address format (xxxx.xxxx.xxxx).

To configure HSRP to use the burned-in MAC address of the interface for the virtual MAC address, use the following command in interface configuration mode:

Command	Purpose
hsrp use-bia [**scope interface**] **Example** : `switch(config-if)# hsrp use-bia`	Configures HSRP to use the burned-in MAC address of the interface for the HSRP virtual MAC address. You can optionally configure HSRP to use the burned-in MAC address for all groups on this interface by using the **scope interface** keywords.

Authenticating HSRP

You can configure HSRP to authenticate the protocol using cleartext or MD5 digest authentication. MD5 authentication uses a key chain.

Before you begin

Ensure that you have enabled the HSRP feature (see the Enabling the HSRP Feature section).

You must configure the same authentication and keys on all members of the HSRP group.

Ensure that you have created the key chain if you are using MD5 authentication.

Procedure

	Command or Action	Purpose

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

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

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

Step 4	**hsrp** *group-number* [**ipv4**]
	Example:
	`switch(config-if)# hsrp 2`
	`switch(config-if-hsrp)`
		Creates an HSRP group and enters HSRP configuration mode.

Step 5	**Option** / **Description** Command / Purpose **authentication text** *string* Example:	Configures cleartext authentication for

	Command or Action		Purpose
	Option	**Description**	
	`switch(config-if-hsrp)# authentication text mypassword`	HSRP on this interface.	
	authentication md5 { **key-chain** *key-chain* | **key-string** { **0** | **7** } **text** [**timeout** *seconds*]} Example: `switch(config-if-hsrp)# authentication md5 key-chain hsrp-keys`	Configures MD5 authentication for HSRP on this interface. You can use a key chain or key string. If you use a key string, you can optionally set the timeout for when HSRP will only accept a new key. The range is from 0 to 32767 seconds.	
Step 6	(Optional) **show hsrp** [**group** *group-number*] **Example:** `switch(config-if-hsrp)# show hsrp group 2`		Displays HSRP information.
Step 7	(Optional) **copy running-config startup-config** **Example:** `switch(config-if-hsrp)# copy running-config startup-config`		Saves this configuration change.

Example

This example shows how to configure MD5 authentication for HSRP on Ethernet 1/2 after creating the key chain:

```
switch# configure terminal
switch(config)# key chain hsrp-keys
switch(config-keychain)# key 0
switch(config-keychain-key)# key-string 7 zqdest
switch(config-keychain-key) accept-lifetime 00:00:00 Jun 01 2008 23:59:59 Sep 12 2008
switch(config-keychain-key) send-lifetime 00:00:00 Jun 01 2008 23:59:59 Aug 12 2008
switch(config-keychain-key) key 1
switch(config-keychain-key) key-string 7 uaeqdyito
switch(config-keychain-key) accept-lifetime 00:00:00 Aug 12 2008 23:59:59 Dec 12 2008
switch(config-keychain-key) send-lifetime 00:00:00 Sep 12 2008 23:59:59 Nov 12 2008
switch(config-keychain-key)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# hsrp 2
```

```
switch(config-if-hsrp)# authenticate md5 key-chain hsrp-keys
switch(config-if-hsrp)# copy running-config startup-config
```

Configuring HSRP Object Tracking

You can configure an HSRP group to adjust its priority based on the availability of other interfaces or routes. The priority of a switch can change dynamically if it has been configured for object tracking and the object that is being tracked goes down. The tracking process periodically polls the tracked objects and notes any value change. The value change triggers HSRP to recalculate the priority. The HSRP interface with the higher priority becomes the active router if you configure the HSRP interface for preemption.

HSRP supports tracked objects and track lists. See Configuring Object Tracking for more information on track lists.

Before you begin

Ensure that you have enabled the HSRP feature (see the Enabling the HSRP Feature section).

Procedure

<table>
<tr><th></th><th colspan="2">Command or Action</th><th>Purpose</th></tr>
<tr><td>Step 1</td><td colspan="2">configure terminal
Example:
<pre>switch# configure terminal
switch(config)#</pre></td><td>Enters configuration mode.</td></tr>
<tr><td>Step 2</td><td colspan="2"><table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td>Command</td><td>Purpose</td></tr>
<tr><td>track object-id interface interface-type number { ip routing | line-protocol }
Example:
<pre>switch(config)# track 1 interface ethernet 2/2 line-protocol
switch(config-track#</pre></td><td>Configures the interface that this HSRP interface tracks. Changes in the state of the interface affect the priority of this HSRP interface as follows:
You configure the interface and corresponding object number that you use with the track command in hsrp configuration mode.</td></tr>
</table></td><td></td></tr>
</table>

<table>
<tr><th></th><th colspan="2">Command or Action</th><th>Purpose</th></tr>
<tr><td></td><td>Option</td><td>Description</td><td></td></tr>
<tr><td></td><td></td><td>• The line-protocol keyword tracks whether the interface is up. The ip keyword also checks that IP routing is enabled on the interface and an IP address is configured.</td><td></td></tr>
<tr><td></td><td>track object-id ip route ip-prefix/length reachability
Example:
<code>switch(config)# track 2 ip route 192.0.2.0/8 reachability
switch(config-track#</code></td><td>Creates a tracked object for a route and enters tracking configuration mode. The object-id range is from 1 to 500.</td><td></td></tr>
<tr><td>Step 3</td><td colspan="2">interface interface-type slot/port
Example:
<code>switch(config)# interface ethernet 1/2
switch(config-if)#</code></td><td>Enters interface configuration mode.</td></tr>
<tr><td>Step 4</td><td colspan="2">no switchport
Example:
<code>switch(config-if)# no switchport</code></td><td>Configures the interface as a Layer 3 routed interface.</td></tr>
<tr><td>Step 5</td><td colspan="2">hsrp group-number [ipv4]
Example:
<code>switch(config-if)# hsrp 2
switch(config-if-hsrp)#</code></td><td>Creates an HSRP group and enters hsrp configuration mode.</td></tr>
<tr><td>Step 6</td><td colspan="2">priority [value]
Example:
<code>switch(config-if-hsrp)# priority 254</code></td><td>Sets the priority level used to select the active router in an HSRP group. The range is from 0 to 255. The default is 100.</td></tr>
<tr><td>Step 7</td><td colspan="2">track object-number [decrement value]
Example:</td><td>Specifies an object to be tracked that affects the weighting of an HSRP interface.</td></tr>
</table>

	Command or Action	Purpose
	`switch(config-if-hsrp)# track 1 decrement 20`	The *value* argument specifies a reduction in the priority of an HSRP interface when a tracked object fails. The range is from 1 to 255. The default is 10.
Step 8	**preempt** [**delay** [**minimum** *seconds*] [**reload** *seconds*] [**sync** *seconds*]] **Example:** `switch(config-if-hsrp)# preempt delay minimum 60`	Configures the router to take over as the active router for an HSRP group if it has a higher priority than the current active router. This command is disabled by default. The range is from 0 to 3600 seconds.
Step 9	(Optional) **show hsrp interface** *interface-type number* **Example:** `switch(config-if-hsrp)# show hsrp interface ethernet 1/2`	Displays HSRP information for an interface.
Step 10	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure HSRP object tracking on Ethernet 1/2:

```
switch# configure terminal
switch(config)# track 1 interface ethernet 2/2 line-protocol
switch(config)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# hsrp 2
switch(config-if-hsrp)# track 1 decrement 20
switch(config-if-hsrp)# copy running-config startup-config
```

Configuring the HSRP Priority

You can configure the HSRP priority on an interface. HSRP uses the priority to determine which HSRP group member acts as the active router.

To configure the HSRP priority, use the following command in interface configuration mode:

Command	Purpose
priority *level* [**forwarding-threshold lower** *lower-value* **upper** *upper-value*] **Example:** `switch(config-if-hsrp)# priority 60 forwarding-threshold lower 40 upper 50`	Sets the priority level used to select the active router in an HSRP group. The *level* range is from 0 to 255. The default is 100.

Customizing HSRP

You can optionally customize the behavior of HSRP. Be aware that as soon as you enable an HSRP group by configuring a virtual IP address, that group is now operational. If you first enable an HSRP group before customizing HSRP, the router could take control over the group and become the active router before you finish customizing the feature. If you plan to customize HSRP, you should do so before you enable the HSRP group.

To customize HSRP, use the following commands in hsrp configuration mode:

Command	Purpose
name *string* **Example:** `switch(config-if-hsrp)# name HSRP-1`	Specifies the IP redundancy name for an HSRP group. The *string* is from 1 to 255 characters. The default string has the following format: **hsrp-interface-short-name**-*group-id* For example, `hsrp-Eth2/1-1`
preempt [**delay** [**minimum** *seconds*] [**reload** *seconds*] [sync *seconds*]] **Example:** `switch(config-if-hsrp)# preempt delay minimum 60`	Configures the router to take over as an active router for an HSRP group if it has a higher priority than the current active router. This command is disabled by default. The range is from 0 to 3600 seconds.
timers [**msec**] *hellotime* [**msec**] *holdtime* **Example:** `switch(config-if-hsrp)# timers 5 18`	Configures the hello and hold time for this HSRP member as follows: The optional *msec* keyword specifies that the argument is expressed in milliseconds, instead of the default seconds. The timer ranges for milliseconds are as follows: • *hellotime* —The interval between successive hello packets sent. The range is from 255 to 999 milliseconds. • *holdtime* —The interval before the information in the hello packet is considered invalid. The range is from 750 to 3000 milliseconds.

To customize HSRP, use the following commands in interface configuration mode:

Command or Action	Purpose
hsrp delay minimum *seconds* **Example:** `switch(config-if)# hsrp delay minimum 30`	Specifies the minimum amount of time that HSRP waits after a group is enabled before participating in the group. The range is from 0 to 10000 seconds. The default is 0.
hsrp delay reload *seconds* **Example:** `switch(config-if)# hsrp delay reload 30`	Specifies the minimum amount of time that HSRP waits after reload before participating in the group. The range is from 0 to 10000 seconds. The default is 0.

Verifying the HSRP Configuration

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

Command	Purpose
show hsrp [**group** *group-number*]	Displays the HSRP status for all groups or one group.
show hsrp delay [**interface** *interface-type slot/port*]	Displays the HSRP delay value for all interfaces or one interface.
show hsrp [**interface** *interface-type slot/port*]	Displays the HSRP status for an interface.
show hsrp [**group** *group-number*] [**interface** *interface-type slot/port*] [**active**] [**all**] [**init**] [**learn**] [**listen**] [**speak**] [**standby**]	Displays the HSRP status for a group or interface for virtual forwarders in the active, init, learn, listen, or standby state. Use the **all** keyword to see all states, including disabled.
show hsrp [**group** *group-number*] [**interface** *interface-type slot/port*] **active**] [**all**] [**init**] [**learn**] [**listen**] [**speak**] [**standby**] **brief**	Displays a brief summary of the HSRP status for a group or interface for virtual forwarders in the active, init, learn, listen, or standby state. Use the **all** keyword to see all states, including disabled.

Configuration Examples for HSRP

This example shows how to enable HSRP on an interface with MD5 authentication and interface tracking:

```
key chain hsrp-keys
key 0
key-string 7 zqdest
accept-lifetime 00:00:00 Jun 01 2008 23:59:59 Sep 12 2008
send-lifetime 00:00:00 Jun 01 2008 23:59:59 Aug 12 2008
key 1
key-string 7 uaeqdyito
accept-lifetime 00:00:00 Aug 12 2008 23:59:59 Dec 12 2008
send-lifetime 00:00:00 Sep 12 2008 23:59:59 Nov 12 2008
feature hsrp
track 2 interface ethernet 2/2 ip
interface ethernet 1/2
no switchport
ip address 192.0.2.2/8
hsrp 1
authenticate md5 key-chain hsrp-keys
priority 90
track 2 decrement 20
ip-address 192.0.2.10
no shutdown
```

This example shows how to configure an HSRP subnet VIP address, which is configured in a different subnet than that of the interface IP address:

```
switch# configure terminal
switch(config)# feature hsrp
switch(config)# feature interface-vlan
switch(config)# interface vlan 2
```

```
switch(config-if)# ip address 192.0.2.1/24
switch(config-if)# hsrp 2
switch(config-if-hsrp)# ip 209.165.201.1/24
```

This example shows a VIP mismatch error when a VIP without a subnet is configured in a different subnet than the interface IP address:

```
switch# configure terminal
switch(config)# feature hsrp
switch(config)# feature interface-vlan
switch(config)# interface vlan 2
switch(config-if)# ip address 192.0.2.1/24
switch(config-if)# hsrp 2
switch(config-if-hsrp)# ip 209.165.201.1
!ERROR: VIP subnet mismatch with interface IP!
```

This example shows a VIP mismatch error when the HSRP subnet VIP address is configured in the same subnet as the interface IP address:

```
switch# configure terminal
switch(config)# feature hsrp
switch(config)# feature interface-vlan
switch(config)# interface vlan 2
switch(config-if)# ip address 192.0.2.1/24
switch(config-if)# hsrp 2
switch(config-if-hsrp)# ip 192.0.2.10/24
!ERROR: Subnet VIP cannot be in same subnet as interface IP!
```

Additional References

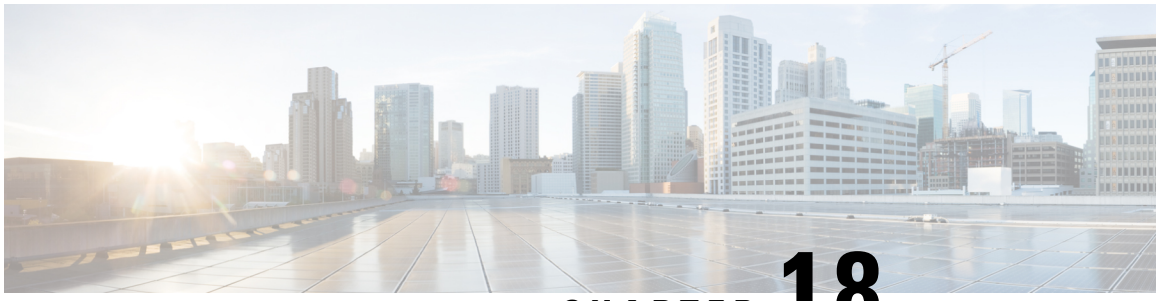
For additional information related to implementing HSRP, see the following sections:

Related Documents

Related Topic	Document Title
Configuring the Virtual Router Redundancy Protocol	Configuring VRRP

MIBs

MIBs	MIBs Link
CISCO-HSRP-MIB	To locate and download MIBs, go to the following: MIB Locator.



CHAPTER 18

Configuring VRRP

This chapter describes how to configure the Virtual Router Redundancy Protocol (VRRP) on Cisco NX-OS switches.

This chapter includes the following sections:

- Information About VRRP, on page 355
- Guidelines and Limitations for VRRP, on page 359
- Default Settings for VRRP, on page 360
- Configuring VRRP, on page 361
- Configuring VRRPv3, on page 369
- Verifying the VRRPv2 Configuration, on page 373
- Verifying the VRRPv3 Configuration, on page 374
- Displaying VRRP Statistics, on page 374
- Configuration Examples for VRRPv2, on page 375
- Configuration Example for VRRPv3, on page 376
- Additional References, on page 376

Information About VRRP

VRRP allows for transparent failover at the first-hop IP router, by configuring a group of routers to share a virtual IP address. VRRP selects a primary router in that group to handle all packets for the virtual IP address. The remaining routers are in standby and take over if that the primary router fails.

VRRP Operation

A LAN client can determine which router should be the first hop to a particular remote destination by using a dynamic process or static configuration. Examples of dynamic router discovery are as follows:

- Proxy ARP—The client uses Address Resolution Protocol (ARP) to get the destination it wants to reach, and a router will respond to the ARP request with its own MAC address.
- Routing protocol—The client listens to dynamic routing protocol updates (for example, from Routing Information Protocol [RIP]) and forms its own routing table.
- ICMP Router Discovery Protocol (IRDP) client—The client runs an Internet Control Message Protocol (ICMP) router discovery client.

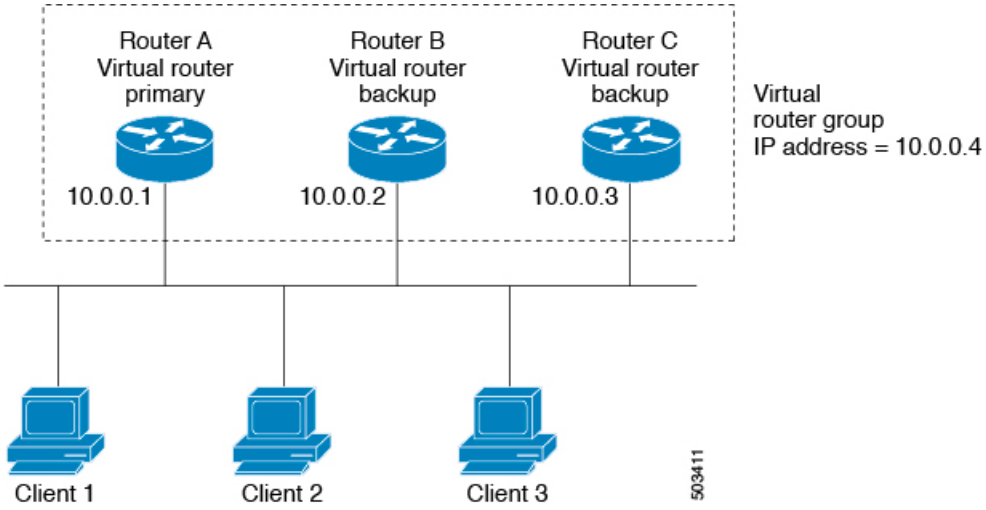
The disadvantage to dynamic discovery protocols is that they incur some configuration and processing overhead on the LAN client. Also, if a router fails, the process of switching to another router can be slow.

An alternative to dynamic discovery protocols is to statically configure a default router on the client. Although this approach simplifies client configuration and processing, it creates a single point of failure. If the default gateway fails, the LAN client is limited to communicating only on the local IP network segment and is cut off from the rest of the network.

VRRP can solve the static configuration problem by enabling a group of routers (a VRRP group) to share a single virtual IP address. You can then configure the LAN clients with the virtual IP address as their default gateway.

The following figure shows a basic VLAN topology. In this example, Routers A, B, and C form a VRRP group. The IP address of the group is the same address that was configured for the Ethernet interface of Router A (10.0.0.1).

Figure 36: Basic VRRP Topology



Because the virtual IP address uses the IP address of the physical Ethernet interface of Router A, Router A is the primary (also known as the IP address owner). As the primary, Router A owns the virtual IP address of the VRRP group router and forwards packets that are sent to this IP address. Clients 1 through 3 are configured with the default gateway IP address of 10.0.0.1.

Routers B and C function as backups. If the primary fails, the backup router with the highest priority becomes the primary and takes over the virtual IP address to provide uninterrupted service for the LAN hosts. When router A recovers, it becomes the router primary again. For more information, see the “VRRP Router Priority and Preemption” section.

Note Packets that are received on a routed port that is destined for the VRRP virtual IP address terminate on the local router, regardless of whether that router is the primary VRRP router or a backup VRRP router. This includes ping and telnet traffic. Packets received on a Layer 2 (VLAN) interface destined for the VRRP virtual IP address will terminate on the primary router.

VRRP Benefits

The benefits of VRRP are as follows:

- Redundance—Enables you to configure multiple routers as the default gateway router, which reduces the possibility of a single point of failure in a network.
- Load Sharing—Allows traffic to and from LAN clients to be shared by multiple routers. The traffic load is shared more equitably among available routers.
- Multiple VRRP groups—Supports up to 255 VRRP groups on a router physical interface if the platform supports multiple MAC addresses. Multiple VRRP groups enable you to implement redundancy and load sharing in your LAN topology.
- Multiple IP Addresses—Allows you to manage multiple IP addresses, including secondary IP addresses. If you have multiple subnets that are configured on an Ethernet interface, you can configure VRRP on each subnet.
- Preemption—Enables you to preempt a backup router that has taken over for a failing primary with a higher priority backup router that has become available.
- Advertisement Protocol—Uses a dedicated Internet Assigned Numbers Authority (IANA) standard multicast address (224.0.0.18) for VRRP advertisements. This addressing scheme minimizes the number of routers that must service the multicasts and allows test equipment to accurately identify VRRP packets on a segment. IANA has assigned the IP protocol number 112 to VRRP.
- The benefits of VRRPv3 are as follows:
  - Interoperability in multivendor environments.
  - Support for the IPv4 and IPv6 address families.

Multiple VRRP Groups

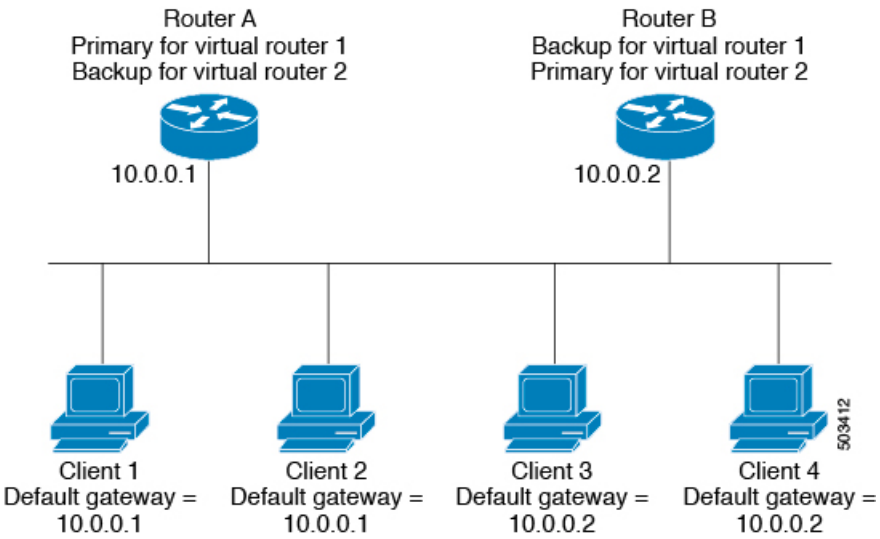
You can configure up to 255 VRRP groups on a physical interface. The actual number of VRRP groups that a router interface can support depends on the following factors:

- Router processing capability
- Router memory capability

In a topology where multiple VRRP groups are configured on a router interface, the interface can act as a primary for one VRRP group and as a backup for one or more other VRRP groups.

The following figure shows a LAN topology in which VRRP is configured so that Routers A and B share the traffic to and from clients 1 through 4. Routers A and B act as backups to each other if either router fails.

Figure 37: Load Sharing and Redundancy VRRP Topology



This topology contains two virtual IP addresses for two VRRP groups that overlap. For VRRP group 1, Router A is the owner of IP address 10.0.0.1 and is the primary. Router B is the backup to router A. Clients 1 and 2 are configured with the default gateway IP address of 10.0.0.1.

For VRRP group 2, Router B is the owner of IP address 10.0.0.2 and is the primary. Router A is the backup to router B. Clients 3 and 4 are configured with the default gateway IP address of 10.0.0.2.

VRRP Router Priority and Preemption

An important aspect of the VRRP redundancy scheme is the VRRP router priority because the priority determines the role that each VRRP router plays and what happens if the primary router fails.

If a VRRP router owns the virtual IP address and the IP address of the physical interface, this router functions as the primary. The priority of the primary is 255.

Priority also determines if a VRRP router functions as a backup router and the order of ascendancy to becoming a primary if the primary fails.

For example, if router A, the primary in a LAN topology fails, VRRP must determine if backups B or C should take over. If you configure router B with priority 101 and router C with the default priority of 100, VRRP selects router B to become the primary because it has the higher priority. If you configure routers B and C with the default priority of 100, VRRP selects the backup with the higher IP address to become the primary.

VRRP uses preemption to determine what happens after a VRRP backup router becomes the primary. With preemption enabled by default, VRRP switches to a backup if that backup comes online with a priority higher than the new primary. For example, if Router A is the primary and fails, VRRP selects Router B (next in order of priority). If Router C comes online with a higher priority than Router B, VRRP selects Router C as the new primary, even though Router B has not failed.

If you disable preemption, VRRP will only switch if the original primary recovers or the new primary fails.

VRRP Advertisements

The VRRP primary sends VRRP advertisements to other VRRP routers in the same group. The advertisements communicate the priority and state of the primary. Cisco NX-OS encapsulates the VRRP advertisements in IP packets and sends them to the IP multicast address assigned to the VRRP group. Cisco NX-OS sends the advertisements once every second by default, but you can configure a different advertisement interval.

VRRP Authentication

VRRP supports the following authentication mechanisms:

- No authentication
- Plain text authentication

VRRP rejects packets in any of the following cases:

- The authentication schemes differ on the router and in the incoming packet.
- Text authentication strings differ on the router and in the incoming packet.

VRRPv3

VRRP version 3 (VRRPv3) enables a group of switches to form a single virtual switch in order to provide redundancy and reduce the possibility of a single point of failure in a network. The LAN clients can then be configured with the virtual switch as their default gateway. The virtual switch, representing a group of switches, is also known as a VRRPv3 group.

Virtualization Support

VRRP supports Virtual Routing and Forwarding instances (VRFs). By default, Cisco NX-OS places you in the default VRF unless you specifically configure another VRF.

If you change the VRF membership of an interface, Cisco NX-OS removes all Layer 3 configuration, including VRRP.

For more information, see Configuring Layer 3 Virtualization.

Guidelines and Limitations for VRRP

VRRP has the following configuration guidelines and limitations:

- You cannot configure VRRP on the management interface.
- When VRRP is enabled, you should replicate the VRRP configuration across switches in your network.
- If you want to perform a graceful failover of the VRRPv3 peer, shut down the protocol first and then shut down the interface. Shutting down the interface does not guarantee a quick transition between the peers before the interface shuts down. Therefore, it can result in a VRRPv3 failover based on the expiry of the hold time.

- We recommend that you do not configure more than one first-hop redundancy protocol on the same interface.
- You must configure an IP address for the interface that you configure VRRP on and enable that interface before VRRP becomes active.
- Cisco NX-OS removes all Layer 3 configurations on an interface when you change the interface VRF membership, port channel membership, or when you change the port mode to Layer 2.
- When you configure VRRP to track a Layer 2 interface, you must shut down the Layer 2 interface and reenable the interface to update the VRRP priority to reflect the state of the Layer 2 interface.
- VRRPv3 has the following configuration guidelines and limitations:
  - VRRPv3 is not intended as a replacement for existing dynamic protocols. VRRPv3 is designed for use over multi-access, multicast, or broadcast-capable Ethernet LANs.
  - VRRPv3 is supported only on Ethernet and Fast Ethernet interfaces, bridge group virtual interfaces (BVIs), and Gigabit Ethernet interfaces as well as on Multiprotocol Label Switching (MPLS) virtual private networks (VPNs), VRF-aware MPLS VPNs, and VLANs.
  - When VRRPv3 is in use, VRRPv2 is unavailable. To configure VRRPv3, you must disable any VRRPv2 configuration.
  - Use VRRPv3 millisecond timers only where absolutely necessary and with careful consideration and testing. Millisecond values work only under favorable circumstances. The millisecond timer values are compatible with third-party vendors, as long as they also support VRRPv3.

Default Settings for VRRP

Table below lists the default settings for VRRP parameters.

Table 24: Default VRRP Parameters

Parameters	Default
advertisement interval	1 seconds
authentication	no authentication
preemption	enabled
priority	100
VRRP feature	disabled
VRRPv3	disabled
VRRPv3 secondary address matchin	Enabled
VRRPv3 timers advertise	1000 milliseconds

Configuring VRRP

Enabling the VRRP Feature

You must globally enable the VRRP feature before you can configure and enable any VRRP groups.

To enable the VRRP feature, use the following command in global configuration mode:

Command	Purpose
feature vrrp **Example** : `switch(config)# feature vrrp`	Enables VRRP.

To disable the VRRP feature and remove all associated configuration, use the following command in global configuration mode:

Command	Purpose
no feature vrrp **Example** : `switch(config)# no feature vrrp`	Disables the VRRP feature.

Configuring VRRP Groups

You can create a VRRP group, assign the virtual IP address, and enable the group.

You can configure one virtual IPv4 address for a VRRP group. By default, the primary VRRP router drops the packets addressed directly to the virtual IP address because the VRRP primary is only intended as a next-hop router to forward packets. Some applications require that Cisco NX-OS accept packets that are addressed to the virtual router IP. Use the secondary option to the virtual IP address to accept these packets when the local router is the VRRP primary.

Once you have configured the VRRP group, you must explicitly enable the group before it becomes active.

Before you begin

Ensure that you configure an IP address on the interface (see the Configuring IPv4 Addressing section).

Procedure

	Command or Action	Purpose

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

	Command or Action	Purpose

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

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

Step 4	**vrrp** *number*
	Example:
	`switch(config-if)# vrrp 250`
	`switch(config-if-vrrp)#`
		Creates a virtual router group. The range is 1–255.

Step 5	**address** *ip-address* [**secondary**] **Example:** `switch(config-if-vrrp)# address 192.0.2.8`	Configures the virtual IPv4 address for the specified VRRP group. This address should be in the same subnet as the IPv4 address of the interface. Use the **secondary** option only if applications require that VRRP routers accept the packets sent to the virtual router's IP address and deliver to applications.

Step 6	**no shutdown**
	Example:
	`switch(config-if-vrrp)# no shutdown`
	`switch(config-if-vrrp)#`
		Enables the VRRP group. Disabled by default.

Step 7	(Optional) **show vrrp** **Example:** `switch(config-if-vrrp)# show vrrp`	Displays VRRP information.
Step 8	(Optional) **copy running-config startup-config** **Example:** `switch(config-if-vrrp)# copy running-config startup-config`	Saves this configuration change.

Configuring VRRP Priority

The valid priority range for a virtual router is 1–254 (1 is the lowest priority and 254 is the highest). The default priority value for backups is 100. For switches whose interface IP address is the same as the primary virtual IP address (the primary), the default value is 255.

Before you begin

Ensure that you have enabled the VRRP feature (see the Configuring VRRP section).

Ensure that you have configured an IP address on the interface (see the Configuring IPv4 Addressing section.

Procedure

	Command or Action	Purpose

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

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

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

Step 4	**vrrp** *number*
	Example:
	`switch(config-if)# vrrp 250`
	`switch(config-if-vrrp)#`
		Creates a virtual router group. The range is 1–255.

Step 5	**shutdown**
	Example:
	`switch(config-if-vrrp)# shutdown`
	`switch(config-if-vrrp)#`
		Disables the VRRP group. Disabled by default.

Step 6	**priority** *level* [**forwarding-threshold lower** *lower-value* **upper** *upper-value*] **Example:** `switch(config-if-vrrp)# priority 60 forwarding-threshold lower 40 upper 50`	Sets the priority level used to select the active router in an VRRP group. The *level* range is 1–254. The default is 100 for backups and 255 for a primary that has an interface IP address equal to the virtual IP address.

Step 7	**no shutdown**
	Example:
	`switch(config-if-vrrp)# no shutdown`
	`switch(config-if-vrrp)#`
		Enables the VRRP group. Disabled by default.

Step 8	(Optional) **show vrrp** **Example:** `switch(config-if-vrrp)# show vrrp`	Displays VRRP information.
Step 9	(Optional) **copy running-config startup-config** **Example:**	Saves this configuration change.

	Command or Action	Purpose
	`switch(config-if-vrrp)# copy running-config startup-config`	

Configuring VRRP Authentication

You can configure simple text authentication for a VRRP group.

Before you begin

Ensure that the authentication configuration is identical for all VRRP switches in the network.

Ensure that you have enabled the VRRP feature (see the Configuring VRRP section).

Ensure that you have configured an IP address on the interface (see the Configuring IPv4 Addressing section.

Procedure

	Command or Action	Purpose

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

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

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

Step 4	**vrrp** *number*
	Example:
	`switch(config-if)# vrrp 250`
	`switch(config-if-vrrp)#`
		Creates a virtual router group. The range is from 1 to 255.

Step 5	**shutdown**
	Example:
	`switch(config-if-vrrp)# shutdown`
	`switch(config-if-vrrp)#`
		Disables the VRRP group. Disabled by default.

Step 6	**authentication text** *password* **Example:** `switch(config-if-vrrp)# authentication text cisco123`	Assigns the simple text authentication option and specifies the keyname password. The keyname range is from 1 to 255 characters. We recommend that you use at least 16 characters.

	Command or Action	Purpose
		The text password is up to eight alphanumeric characters.

Step 7	**no shutdown**
	Example:
	`switch(config-if-vrrp)# no shutdown`
	`switch(config-if-vrrp)#`
		Enables the VRRP group. Disabled by default.

Step 8	(Optional) **show vrrp** **Example:** `switch(config-if-vrrp)# show vrrp`	Displays VRRP information.
Step 9	(Optional) **copy running-config startup-config** **Example:** `switch(config-if-vrrp)# copy running-config startup-config`	Saves this configuration change.

Example

Configuring Time Intervals for Advertisement Packets

You can configure the time intervals for advertisement packets.

Before you begin

Ensure that you have enabled the VRRP feature (see the Configuring VRRP section).

Ensure that you have configured an IP address on the interface (see the Configuring IPv4 Addressing section.

Procedure

	Command or Action	Purpose

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

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

Step 3	**no switchport** **Example:**	Configures the interface as a Layer 3 routed interface.

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

Step 4	**vrrp** *number*
	Example:
	`switch(config-if)# vrrp 250`
	`switch(config-if-vrrp)#`
		Creates a virtual router group. The range is from 1 to 255.

Step 5	**shutdown**
	Example:
	`switch(config-if-vrrp)# shutdown`
	`switch(config-if-vrrp)#`
		Disables the VRRP group. Disabled by default.

Step 6	**advertisement-interval** *seconds*
	Example:
	`switch(config-if-vrrp)#`
	`advertisement-interval 15`
		Sets the interval time in seconds between sending advertisement frames. The range is from 1 to 254. The default is 1 second.

Step 7	**no shutdown**
	Example:
	`switch(config-if-vrrp)# no shutdown`
	`switch(config-if-vrrp)#`
		Enables the VRRP group. Disabled by default.

Step 8	(Optional) **show vrrp** **Example:** `switch(config-if-vrrp)# show vrrp`	Displays VRRP information.

Step 9	(Optional) **copy running-config startup-config**
	Example:
	`switch(config-if-vrrp)# copy`
	`running-config startup-config`
		Saves this configuration change.

Example

Disabling Preemption

You can disable preemption for a VRRP group member. If you disable preemption, a higher-priority backup router will not take over for a lower-priority primary router. Preemption is enabled by default.

Before you begin

Ensure that you have enabled the VRRP feature (see the Configuring VRRP section).

Ensure that you have configured an IP address on the interface (see the Configuring IPv4 Addressing section.

Procedure

	Command or Action	Purpose

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

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

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

Step 4	**vrrp** *number*
	Example:
	`switch(config-if)# vrrp 250`
	`switch(config-if-vrrp)#`
		Creates a virtual router group. The range is 1–255.

Step 5	**shutdown**
	Example:
	`switch(config-if-vrrp)# shutdown`
	`switch(config-if-vrrp)#`
		Disables the VRRP group. Disabled by default.

Step 6	**no preempt** **Example:** `switch(config-if-vrrp)# no preempt`	Disables the preempt option and allows the primary to remain when a higher-priority backup appears.

Step 7	**no shutdown**
	Example:
	`switch(config-if-vrrp)# no shutdown`
	`switch(config-if-vrrp)#`
		Enables the VRRP group. Disabled by default.

Step 8	(Optional) **show vrrp** **Example:** `switch(config-if-vrrp)# show vrrp`	Displays VRRP information.
Step 9	(Optional) **copy running-config startup-config** **Example:** `switch(config-if-vrrp)# copy running-config startup-config`	Saves this configuration change.

Configuring VRRP Interface State Tracking

Interface state tracking changes the priority of the virtual router based on the state of another interface in the switch. When the tracked interface goes down or the IP address is removed, Cisco NX-OS assigns the tracking priority value to the virtual router. When the tracked interface comes up and an IP address is configured on this interface, Cisco NX-OS restores the configured priority to the virtual router (see the Configuring VRRP Priority section).

Note For interface state tracking to function, you must enable preemption on the interface.

Note VRRP does not support Layer 2 interface tracking.

Before you begin

Ensure that you have enabled the VRRP feature (see the Configuring VRRP section).

Ensure that you have configured an IP address on the interface (see the Configuring IPv4 Addressing section.

Be sure the virtual router is enabled (see the Configuring VRRP Groups section).

Procedure

	Command or Action	Purpose

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

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

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

Step 4	**vrrp** *number*
	Example:
	`switch(config-if)# vrrp 250`
	`switch(config-if-vrrp)#`
		Creates a virtual router group. The range is from 1 to 255.

Step 5	**shutdown** **Example:**	Disables the VRRP group. Disabled by default.

	Command or Action	Purpose
	`switch(config-if-vrrp)# shutdown` `switch(config-if-vrrp)#`	
Step 6	**track interface type** *number* **priority** *value* **Example:** `switch(config-if-vrrp)# track interface ethernet 2/10 priority 254`	Enables interface priority tracking for a VRRP group. The priority range is from 1 to 254.

Step 7	**no shutdown**
	Example:
	`switch(config-if-vrrp)# no shutdown`
	`switch(config-if-vrrp)#`
		Enables the VRRP group. Disabled by default.

Step 8	(Optional) **show vrrp** **Example:** `switch(config-if-vrrp)# show vrrp`	Displays VRRP information.
Step 9	(Optional) **copy running-config startup-config** **Example:** `switch(config-if-vrrp)# copy running-config startup-config`	Saves this configuration change.

Configuring VRRPv3

Enabling VRRPv3

You must globally enable the VRRPv3 feature before you can configure and enable any VRRPv3 groups.

To enable the VRRPv3 feature, use the following command in global configuration mode:

Command	Purpose
feature vrrpv3 **Example** : `switch(config)# feature vrrpv3`	Enables VRRP version 3. The no form of this command disables VRRPv3 in a VDC. If VRRPv2 is currently configured, use the no feature vrrp command in global configuration mode to remove the VRRPv2 configuration and then use the feature vrrpv3 command to enable VRRPv3.

Configuring VRRPv3 Groups

You can create a VRRPv3 group, assign the virtual IP address, and enable the group.

You can configure one virtual IPv4 address for a VRRPv3 group. By default, the primary VRRPv3 router drops the packets addressed directly to the virtual IP address because the VRRPv3 primary is only intended as a next-hop router to forward packets. Some applications require that Cisco NX-OS accept packets that are

addressed to the virtual router IP. Use the secondary option to the virtual IP address to accept these packets when the local router is the VRRPv3 primary.

Note After you have configured the VRRPv3 group, you must explicitly enable the group before it becomes active.

Before you begin

- Ensure that the VRRPv3 feature is enabled.
- Ensure that you configure an IP address on the interface.

Procedure

	Command or Action	Purpose

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

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

Step 3	**[no] vrrpv3 number address-family { ipv4 | ipv6 }**
	Example:
	`switch(config-if)# vrrpv3 5 address-family`
	`ipv4`
	`switch(config-if-vrrpv3-group)#`
		Creates a VRRPv3 group and enters VRRPv3 group configuration mode. The range is 1–255.
		The no form of this command removes all configuration that is defined within the submode.

Step 4	(Optional) **[no] address** *ip-address* [**primary** | **secondary**]
	Example:
	`switch(config-if-vrrpv3-group)# address`
	`100.0.1.10 primary`
		Specifies a primary or secondary IPv4 or IPv6 address for the VRRPv3 group.
		Note To utilize secondary IP addresses in a VRRPv3 group, you must first configure a primary IP address on the same group.

Step 5	(Optional) **[no] description** *description*
	Example:
	`switch(config-if-vrrpv3-group)# description`
	`group3`
		Specifies a description for the VRRPv3 group. You can enter up to 80 alphanumeric characters.

	Command or Action	Purpose
Step 6	(Optional) **[no] match-address** **Example:** `switch(config-if-vrrpv3-group)# match-address`	Matches the secondary address in the advertisement packet against the configured address.
Step 7	(Optional) **[no] preempt** [**delay minimum** *seconds*] **Example:** `switch(config-if-vrrpv3-group)# preempt delay minimum 30`	Enables preemption of a lower priority primary switch with an optional delay. The range is 0–3600.
Step 8	(Optional) **[no] priority** *level* **Example:** `switch(config-if-vrrpv3-group)# priority 3`	Specifies the priority of the VRRPv3 group. The range is 1–254.
Step 9	(Optional) **[no] timers advertise** *interval* **Example:** `switch(config-if-vrrpv3-group)# timers advertise 1000`	Sets the advertisement timer in milliseconds. The range is 100–40950. **Note** Cisco recommends that you set this timer to a value greater than or equal to 1 second.
Step 10	**[no] vrrpv2** **Example:** `switch(config-if-vrrpv3-group)# vrrpv2`	Enables support for VRRPv2 simultaneously, to ensure interoperability with devices that support only VRRPv2. **Note** VRRPv2 compatibility mode is provided to allow an upgrade from VRRPv2 to VRRPv3. This is not a full VRRPv2 implementation and should be used only to perform an upgrade.
Step 11	(Optional) **[no] shutdown** **Example:** `switch(config-if-vrrp3-group)# shutdown`	Disables VRRP configuration for the VRRPv3 group.
Step 12	(Optional) **show fhrp** [*interface-type interface-number*] [**verbose**] **Example:** `switch(config-if-vrrp3-group)# show fhrp port-channnel 101 verbose`	Displays First Hop Redundancy Protocol (FHRP) information. Use the **verbose** keyword to view detailed information.
Step 13	(Optional) **copy running-config startup-config** **Example:**	Saves this configuration change.

	Command or Action	Purpose
	`switch(config-if-vrrp3-group)# copy running-config startup-config`	

Configuring the Delay Period for FHRP Client Initialization

You can configure the delay period for the initialization of FHRP clients.

Note In all FHRP protocols, we do not recommend to use aggressive timers as they cause CPU spikes and they result in increased control packet flow. In case of VRRPv3, you should configure sufficient interface delay/reload delay for proper failover of the VRRP nodes.

To configure this feature, use the following command in interface configuration mode:

Command	Purpose
fhrp delay {[minimum] | [reload] *seconds* } **Example** : `switch(config)# fhrp delay minimum 34`	Specifies the delay period for the initialization of FHRP clients. The range is from 0 to 3600 seconds. The **minimum** keyword configures the delay period after an interface becomes available. The **reload** command configures the delay period after the device reloads.

Configuring VRRPv3 Control Groups

You can configure a VRRPv3 control group.

Before you begin

- Ensure that the VRRPv3 feature is enabled.
- Ensure that you configure an IP address on the interface.

Procedure

	Command or Action	Purpose

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

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

	Command or Action	Purpose
Step 3	**[no] ip address ip-address mask [secondary]** **Example:** `switch(config-if)# ip address 209.165.200.230 255.255.255.224`	Configures the IP address on the interface. **Note** You can use the **secondary** keyword to configure additional IP addresses on the interface.
Step 4	**[no] vrrpv3 number address-family { ipv4 | ipv6 }** **Example:** `switch(config-if)# vrrpv3 5 address-family ipv4 switch(config-if-vrrpv3-group)#`	Creates a VRRPv3 group and enters VRRPv3 group configuration mode. The range is from 1 to 255.
Step 5	(Optional) **[no] address ip-address [primary | secondary]** **Example:** `switch(config-if-vrrpv3-group)# address 209.165.200.227 primary`	Specifies a primary or secondary IPv4 or IPv6 address for the VRRPv3 group.
Step 6	(Optional) **[no] shutdown** **Example:** `switch(config-if-vrrp3-group)# shutdown`	Disables VRRP configuration for the VRRPv3 group.
Step 7	[**show fhrp** [*interface-type interface-number*] [**verbose**] **Example:** `switch(config-if-vrrp3-group)# show fhrp port-channnel 101 verbose`	Displays First Hop Redundancy Protocol (FHRP) information. Use the **verbose** keyword to view detailed information.
Step 8	**copy running-config startup-config** **Example:** `switch(config-if-vrrp3-group)# copy running-config startup-config`	Saves this configuration change.

Verifying the VRRPv2 Configuration

To display the VRRPv2 configuration information, perform one of the following tasks:

Command	Purpose
show vrrpv2	Displays the VRRP status for all groups.
show vrrpv2 vr group-number	Displays the VRRP status for a VRRP group.
show vrrp v2 vr number interface interface-type port configuration	Displays the virtual router configuration for an interface.

Command	Purpose
show vrrpv2 vr number interface interface-type portstatus	Displays the virtual router status for an interface.
show fhrp [*interface-type interface-number*] [**verbose**]	Displays First Hop Redundancy Protocol (FHRP) information.
show interface *interface-type*	Displays the virtual router configuration for an interface.

Verifying the VRRPv3 Configuration

See the following table for information on the fields in the show vrrpv3 command output:

Command	Purpose
>	Redirect it to a file
>>	Redirect it to a file in append mode
all	All VRRPV3 information
brief	Brief output
detail	Detail output
ethernet	Ethernet IEEE 802.3z
ipv4	IPv4
ipv6	IPv6
loopback	Loopback interface
port-channel	Port-channel interface
statistics	Statistics output
vlan	VLAN interface
|	Pipe command output to filter

For example, use the show vrrpv3 statistics command to display the VRRPv3 statistics.

Displaying VRRP Statistics

To display VRRP statistics, use the following commands:

Command	Purpose
show vrrp statistics interface *interface-type port vr number*	Displays the virtual router information.

Command	Purpose
show vrrp statistics	Displays the VRRP statistics.

Use the **clear vrrp vr** command to clear the IPv4 VRRP statistics for a specified interface.

Use the **clear vrrp ipv4** command to clear all the statistics for the specified IPv4 virtual router.

Configuration Examples for VRRPv2

In this example, Router A and Router B each belong to three VRRP groups. In the configuration, each group has the following properties:

- Group 1:
  - Virtual IP address is 10.1.0.10.
  - Router A becomes the primary for this group with priority 120.
  - Advertising interval is 3 seconds.
  - Preemption is enabled.
- Group 5:
  - Router B becomes the primary for this group with priority 200.
  - Advertising interval is 30 seconds.
  - Preemption is enabled.
- Group 100:
  - Router A becomes the primary for this group first because it has a higher IP address (10.1.0.2).
  - Advertising interval is the default 1 second.
  - Preemption is disabled.

Router A

```
interface ethernet 1/0
no switchport

ip address 10.1.0.2/16
no shutdown
vrrpv2 1
priority 120
authentication text cisco
advertisement-interval 3
address 10.1.0.10
no shutdown
vrrpv2 5
priority 100
advertisement-interval 30
address 10.1.0.50
no shutdown
```

```
vrrpv2 100
no preempt
address 10.1.0.100
no shutdown
```

Router B

```
interface ethernet 1/0
no switchport

ip address 10.2.0.1/2
no shutdown
vrrpv2 1
priority 100
authentication text cisco
advertisement-interval 3
address 10.2.0.10
no shutdown

vrrpv2 5
priority 200
advertisement-interval 30
address 10.2.0.50
no shutdown
vrrpv2 100
no preempt
address 10.2.0.100
no shutdown
```

Configuration Example for VRRPv3

See the following configuration example for VRRPv3.

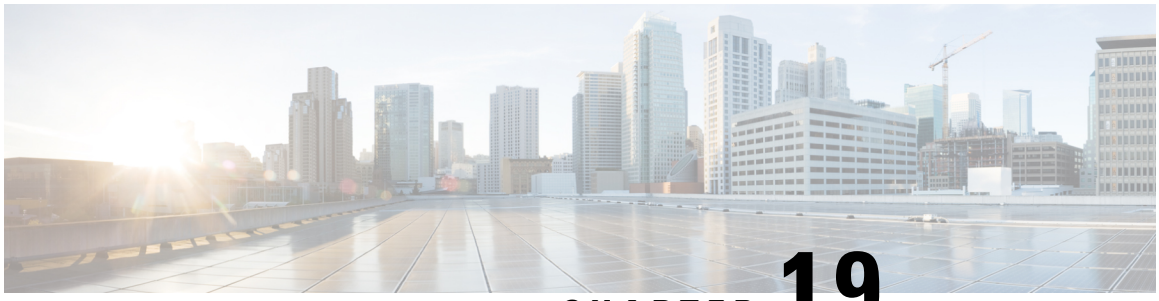
```
interface Vlan20
vrrpv3 10 address-family ipv4
timers advertise 1000
priority 100
preempt
match-address
no vrrpv2
address 20.1.1.1 primary
address 20.1.1.5 secondary
vrrpv3 10 address-family ipv6
timers advertise 1000
priority 100
preempt
match-address
no vrrpv2
address fe80::1 primary
address 2011::5
```

Additional References

For additional information related to implementing VRRP, see the following sections:

Related Documents

Related Topic	Document Title
Configuring the Hot Standby Routing Protocol	Configuring HSRP



CHAPTER 19

Configuring Object Tracking

This chapter describes how to configure object tracking on Cisco NX-OS switches.

This chapter includes the following sections:

- Information About Object Tracking, on page 379
- Guidelines and Limitations for Object Tracking, on page 380
- Default Settings for Object Tracking, on page 381
- Configuring Object Tracking, on page 381
- Verifying the Object Tracking Configuration, on page 389
- Configuration Examples for Object Tracking, on page 389
- Related Topics, on page 389
- Additional References, on page 390
- IETF RFCs, on page 390

Information About Object Tracking

Object tracking allows you to track specific objects on the switch, such as the interface line protocol state, IP routing, and route reachability, and to take action when the tracked object's state changes. This feature allows you to increase the availability of the network and shorten recovery time if an object state goes down.

Object Tracking Overview

The object tracking feature allows you to create a tracked object that multiple clients can use to modify the client behavior when a tracked object changes. Several clients register their interest with the tracking process, track the same object, and take different actions when the object state changes.

Clients include the following features:

- Hot Standby Redundancy Protocol (HSRP)
- Virtual Router Redundancy Protocol (VRRP)
- Embedded Event Manager (EEM)

The object tracking monitors the status of the tracked objects and communicates any changes made to interested clients. Each tracked object is identified by a unique number that clients can use to configure the action to take when a tracked object changes state.

Cisco NX-OS tracks the following object types:

- Interface line protocol state—Tracks whether the line protocol state is up or down.
- Interface IP routing state—Tracks whether the interface has an IPv4 address and if IPv4 routing is enabled and active.
- IP route reachability—Tracks whether an Ipv4 route exists and is reachable from the local switch.

For example, you can configure HSRP to track the line protocol of the interface that connects one of the redundant routers to the rest of the network. If that link protocol goes down, you can modify the priority of the affected HSRP router.

Object Track List

An object track list allows you to track the combined states of multiple objects. Object track lists support the following capabilities:

- Boolean "and" function—Each object defined within the track list must be in an up state so that the track list object can become up.
- Boolean "or" function—At least one object defined within the track list must be in an up state so that the tracked object can become up.
- Threshold percentage—The percentage of up objects in the tracked list must be greater than the configured up threshold for the tracked list to be in the up state. If the percentage of down objects in the tracked list is above the configured track list down threshold, the tracked list is marked as down.
- Threshold weight—Assign a weight value to each object in the tracked list, and a weight threshold for the track list. If the combined weights of all up objects exceeds the track list weight up threshold, the track list is in an up state. If the combined weights of all the down objects exceeds the track list weight down threshold, the track list is in the down state.

See the Configuring an Object Track List with a Boolean Expression section for more information on track lists.

Virtualization Support

Object tracking supports Virtual Routing and Forwarding (VRF) instances. By default, Cisco NX-OS places you in the default VRF unless you specifically configure another VRF. By default, Cisco NX-OS tracks the route reachability state of objects in the default VRF. If you want to track objects in another VRF, you must configure the object to be a member of that VRF (see the Configuring Object Tracking for a Nondefault VRF section).

For more information, see Configuring Layer 3 Virtualization.

Guidelines and Limitations for Object Tracking

Object tracking has the following configuration guidelines and limitations:

- Supports up to 500 tracked objects.
- Supports Ethernet, subinterfaces, tunnels, port channels, loopback interfaces, and VLAN interfaces.

- Supports one tracked object per HSRP group.

Default Settings for Object Tracking

Table below lists the default settings for object tracking parameters.

Table 25: Default Object Tracking Parameters

Parameters	Default
Tracked Object VRF	Member of default VRF

Configuring Object Tracking

Configuring Object Tracking for an Interface

You can configure Cisco NX-OS to track the line protocol or IPv4 routing state of an interface.

Procedure

	Command or Action	Purpose

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

Step 2	**track** *object-id* **interface** *interface-type number* { **ip routing** | **line-protocol** }
	Example:
	`switch(config)# track 1 interface ethernet 1/2 line-protocol`
	`switch(config-track#`
		Creates a tracked object for an interface and enters tracking configuration mode. The *object-id* range is from 1 to 500.

Step 3	(Optional) **show track** [*object-id*] **Example:** `switch(config-track)# show track 1`	Displays object tracking information
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config-track)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure object tracking for the line protocol state on Ethernet 1/2:

```
switch# configure terminal
switch(config)# track 1 interface ethernet 1/2 line-protocol
switch(config-track)# copy running-config startup-config
```

This example shows how to configure object tracking for the IPv4 routing state on Ethernet 1/2:

```
switch# configure terminal
switch(config)# track 2 interface ethernet 1/2 ip routing
switch(config-track)# copy running-config startup-config
```

Configuring Object Tracking for Route Reachability

You can configure Cisco NX-OS to track the existence and reachability of an IP route.

Procedure

	Command or Action	Purpose

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

Step 2	**track** *object-id* **ip route** *prefix/length* **reachability**
	Example:
	`switch(config)# track 2 ip route 192.0.2.0/8 reachability`
	`switch(config-track)#`
		Creates a tracked object for a route and enters tracking configuration mode. The *object-id* range is from 1 to 500. The prefix format for IP is A.B.C.D/length, where the length range is from 1 to 32.

Step 3	(Optional) **show track** [*object-id*] **Example:** `switch(config-track)# show track 1`	Displays object tracking information
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config-track)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure object tracking for an IPv4 route in the default VRF:

```
switch# configure terminal
switch(config)# track 4 ip route 192.0.2.0/8 reachability
switch(config-track)# copy running-config startup-config
```

Configuring an Object Track List with a Boolean Expression

You can configure an object track list that contains multiple tracked objects. A tracked list contains one or more objects. The Boolean expression enables two types of calculation by using either "and" or "or" operators. For example, when tracking two interfaces using the "and" operator, up means that both interfaces are up, and down means that either interface is down.

Procedure

	Command or Action	Purpose

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

Step 2	**track** *track-number* **list boolean** { **and** | **or** }
	Example:
	`switch(config)# track 1 list boolean and`
	`switch(config-track#`
		Configures a tracked list object and enters tracking configuration mode. Specifies that the state of the tracked list is based on a Boolean calculation. The keywords are as follows:
		• and —Specifies that the list is up if all objects are up, or down if one or more objects are down. For example, when tracking two interfaces, up means that both interfaces are up, and down means that either interface is down.
		• **or** —Specifies that the list is up if at least one object is up. For example, when tracking two interfaces, up means that either interface is up, and down means that both interfaces are down.
		The *track-number* range is from 1 to 500.

Step 3	**object** *object-id* [**not**] **Example:** `switch(config-track)# object 10`	Adds a tracked object to the track list. The *object-id* range is from 1 to 500. The **not** keyword optionally negates the tracked object state. **Note** The example means that when object 10 is up, the tracked list detects object 10 as down.
Step 4	(Optional) **show track** **Example:** `switch(config-track)# show track`	Displays object tracking information
Step 5	(Optional) **copy running-config startup-config** **Example:**	Saves this configuration change.

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

Example

This example shows how to configure a track list with multiple objects as a Boolean “and”:

```
switch# configure terminal
switch(config)# track 1 list boolean and
switch(config-track)# object 10
switch(config-track)# object 20 not
```

Configuring an Object Track List with a Percentage Threshold

You can configure an object track list that contains a percentage threshold. A tracked list contains one or more objects. The percentage of up objects must exceed the configured track list up percent threshold before the track list is in an up state. For example, if the tracked list has three objects, and you configure an up threshold of 60 percent, two of the objects must be in the up state (66 percent of all objects) for the track list to be in the up state.

Procedure

	Command or Action	Purpose

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

Step 2	**track** *track-number* **list threshold percentage**
	Example:
	`switch(config)# track 1 list threshold percentage`
	`switch(config-track#`
		Configures a tracked list object and enters tracking configuration mode. Specifies that the state of the tracked list is based on a configured threshold percent.
		The *track-number* range is from 1 to 500.

Step 3	**threshold percentage up** *up-value* **down** *down-value* **Example:** `switch(config-track)# threshold percentage up 70 down 30`	Configures the threshold percent for the tracked list. The range from 0 to 100 percent.
Step 4	(Optional) **object** [*object-id*] **Example:** `switch(config-track)# object 10`	Adds a tracked object to the track list. The *object-id* range is from 1 to 500.
Step 5	(Optional) **show track** **Example:** `switch(config-track)# show track`	Displays object tracking information

	Command or Action	Purpose
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-track)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure a track list with an up threshold of 70 percent and a down threshold of 30 percent:

```
switch# configure terminal
switch(config)# track 1 list threshold percentage
switch(config-track)# threshold percentage up 70 down 30
switch(config-track)# object 10
switch(config-track)# object 20
switch(config-track)# object 30
```

Configuring an Object Track List with a Weight Threshold

You can configure an object track list that contains a weight threshold. A tracked list contains one or more objects. The combined weight of up objects must exceed the configured track list up weight threshold before the track list is in an up state. For example, if the tracked list has three objects with the default weight of 10 each, and you configure an up threshold of 15, two of the objects must be in the up state (combined weight of 20) for the track list to be in the up state.

Procedure

	Command or Action	Purpose

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

Step 2	**track** *track-number* **list threshold weight**
	Example:
	`switch(config)# track 1 list threshold weight`
	`switch(config-track#`
		Configures a tracked list object and enters tracking configuration mode. Specifies that the state of the tracked list is based on a configured threshold weight.
		The *track-number* range is from 1 to 500.

Step 3	**threshold weight up** *up-value* **down** *down-value* **Example:** `switch(config-track)# threshold weight up 30 down 10`	Configures the threshold weight for the tracked list. The range from 1 to 255.
Step 4	**object** *object-id* **weight** *value* **Example:**	Adds a tracked object to the track list. The *object-id* range is from 1 to 500. The *value*

	Command or Action	Purpose
	`switch(config-track)# object 10 weight 15`	range is from 1 to 255. The default weight value is 10.
Step 5	(Optional) **show track** **Example:** `switch(config-track)# show track`	Displays object tracking information
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-track)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure a track list with an up weight threshold of 30 and a down threshold of 10:

```
switch# configure terminal
switch(config)# track 1 list threshold weight
switch(config-track)# threshold weight up 30 down 10
switch(config-track)# object 10 weight 15
switch(config-track)# object 20 weight 15
switch(config-track)# object 30
```

In this example, the track list is up if object 10 and object 20 are up, and the track list goes to the down state if all three objects are down.

Configuring an Object Tracking Delay

You can configure a delay for a tracked object or an object track list that delays when the object or list triggers a stage change. The tracked object or track list starts the delay timer when a state change occurs but does not recognize a state change until the delay timer expires. At that point, Cisco NX-OS checks the object state again and records a state change only if the object or list currently has a changed state. Object tracking ignores any intermediate state changes before the delay timer expires.

For example, for an interface line-protocol tracked object that is in the up state with a 20-second down delay, the delay timer starts when the line protocol goes down. The object is not in the down state unless the line protocol is down 20 seconds later.

You can configure independent up delay and down delay for a tracked object or track list. When you delete the delay, object tracking deletes both the up and down delay.

You can change the delay at any point. If the object or list is already counting down the delay timer from a triggered event, the new delay is computed as the following:

- If the new configuration value is less than the old configuration value, the timer starts with the new value .
- If the new configuration value is more than the old configuration value, the timer is calculated as the new configuration value minus the current timer countdown minus the old configuration value.

Procedure

	Command or Action	Purpose

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

Step 2	track *object- id* { *parameters* }
	Example:
	`switch(config)# track 2 ip route 192.0.2.0/8 reachability`
	`switch(config-track)#`
		Creates a tracked object for a route and enters tracking configuration mode. The *object-id* range is from 1 to 500. The prefix format for IP is A.B.C.D/length, where the length range is from 1 to 32.

Step 3	**track** *track-number* **list**{ *parameters* }
	Example:
	`switch(config)# track 1 list threshold weight`
	`switch(config-track#`
		Configures a tracked list object and enters tracking configuration mode. Specifies that the state of the tracked list is based on a configured threshold weight.
		The *track-number* range is from 1 to 500.

Step 4	**delay** { **up** *up-time* [**down** *down-time*] | **down** *down-time* [**up** *up-time*]} **Example:** `switch(config-track)# delay up 20 down 30`	Configures the object delay timers. The range is from 0 to 180 seconds.
Step 5	(Optional) **show track** **Example:** `switch(config-track)# show track`	Displays object tracking information
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-track)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure object tracking for a route and use delay timers:

```
switch# configure terminal
switch(config)# track 2 ip route 209.165.201.0/8 reachability
switch(config-track)# delay up 20 down 30
switch(config-track)# copy running-config startup-config
```

This example shows how to configure a track list with an up weight threshold of 30 and a down threshold of 10 with delay timers:

```
switch# configure terminal
switch(config)# track 1 list threshold weight
switch(config-track)# threshold weight up 30 down 10
switch(config-track)# object 10 weight 15
```

```
switch(config-track)# object 20 weight 15
switch(config-track)# object 30
switch(config-track)# delay up 20 down 30
```

This example shows the delay timer in the show track command output before and after an interface is shut down:

```
switch(config-track)# show track
Track 1
Interface loopback1 Line Protocol
Line Protocol is UP
1 changes, last change 00:00:13
Delay down 10 secs

switch(config-track)# interface loopback 1
switch(config-if)# shutdown
switch(config-if)# show track
Track 1
Interface loopback1 Line Protocol
Line Protocol is delayed DOWN (8 secs remaining)<------- delay timer counting down
1 changes, last change 00:00:22
Delay down 10 secs
```

Configuring Object Tracking for a Nondefault VRF

You can configure Cisco NX-OS to track an object in a specific VRF.

Procedure

	Command or Action	Purpose

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

Step 2	**track object-id ip route** *prefix/length* **reachability**
	Example:
	`switch(config)# track 2 ip route 192.0.2.0/8 reachability`
	`switch(config-track)#`
		Creates a tracked object for a route and enters tracking configuration mode. The *object-id* range is from 1 to 500. The prefix format for IP is A.B.C.D/length, where the length range is from 1 to 32.

Step 3	**vrf member** *vrf-name* **Example:** `switch(config-track)# vrf member Red`	Configures the VRF to use for tracking the configured object.
Step 4	(Optional) **show track** [*object-id*] **Example:** `switch(config-track)# show track 3`	Displays object tracking information
Step 5	(Optional) **copy running-config startup-config** **Example:**	Saves this configuration change.

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

Example

This example shows how to configure object tracking for a route and use VRF Red to look up reachability information for this object:

```
switch# configure terminal
switch(config)# track 2 ip route 209.165.201.0/8 reachability
switch(config-track)# vrf member Red
switch(config-track)# copy running-config startup-config
```

This example shows how to modify tracked object 2 to use VRF Blue instead of VRF Red to look up reachability information for this object:

```
switch# configure terminal
switch(config)# track 2
switch(config-track)# vrf member Blue
switch(config-track)# copy running-config startup-config
```

Verifying the Object Tracking Configuration

To display the object tracking configuration information, perform one of the following tasks:

Command	Purpose
show track [*object-id*] [**brief**]	Displays the object tracking information for one or more objects.
show track [*object-id*] **interface** [**brief**]	Displays the interface-based object tracking information.
show track [*object-id*] **ip-route** [**brief**]	Displays the IPv4 route-based object tracking information.

Configuration Examples for Object Tracking

This example shows how to configure object tracking for route reachability and use VRF Red to look up reachability information for this route:

```
switch# configure terminal
switch(config)# track 2 ip route 209.165.201.0/8 reachability
switch(config-track)# vrf member Red
switch(config-track)# copy running-config startup-config
```

Related Topics

See the following topics for information related to object tracking:

- Configuring Layer 3 Virtualization
- Configuring HSRP

Additional References

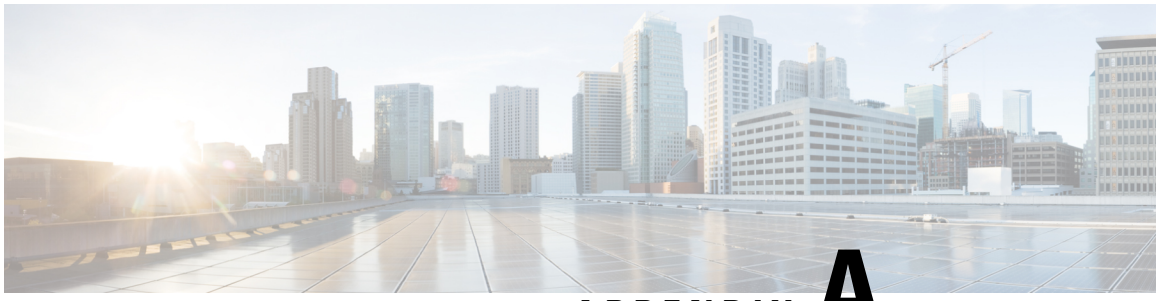
For additional information related to implementing object tracking, see the following sections:

Related Documents

Related Topic	Document Title
Configuring the Embedded Event Manager	Cisco Nexus 3600 System Management Configuration Guide

IETF RFCs

This appendix lists the supported IETF RFCs.



APPENDIX A

IETF RFCs

This appendix lists the supported IETF RFCs.

- IETF RFCs, on page 391

IETF RFCs

BGP RFCs

RFCs	Title
RFC 1997	*BGP Communities Attribute*
RFC 2385	*Protection of BGP Sessions via the TCP MD5 Signature Option*
RFC 2439	*BGP Route Flap Damping*
RFC 2519	*A Framework for Inter-Domain Route Aggregation*
RFC 2858	*Multiprotocol Extensions for BGP-4*
RFC 3065	*Autonomous System Confederations for BGP*
RFC 3392	*Capabilities Advertisement with BGP-4*
RFC 4271	*A Border Gateway Protocol 4 (BGP-4)*
RFC 4273	*Definitions of Managed Objects for BGP-4*
RFC 4456	*BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)*
RFC 4486	*Subcodes for BGP Cease Notification Message*
RFC 4893	*BGP Support for Four-octet AS Number Space*
RFC 5004	*Avoid BGP Best Path Transitions from One External to Another*
draft-ietf-idr-bgp4-mib-15.txt	*BGP4-MIB*

First-Hop Redundancy Protocols RFCs

RFCs	Title
RFC 2281	*Hot Standby Redundancy Protocol*
RFC 3768	*Virtual Router Redundancy Protocol*

IP Services RFCs

RFCs	Title
RFC 786	*UDP*
RFC 791	*IP*
RFC 792	*ICMP*
RFC 793	*TCP*
RFC 826	*ARP*
RFC 1027	*Proxy ARP*
RFC 1591	*DNS Client*
RFC 1812	*IPv4 routers*

OSPF RFCs

RFCs	Title
RFC 2328	*OSPF Version 2*
RFC 3101	*The OSPF Not-So-Stubby Area (NSSA) Option*
RFC 2370	*The OSPF Opaque LSA Option*
RFC 3137	*OSPF Stub Router Advertisement*

RIP RFCs

RFCs	Title
RFC 2453	*RIP Version 2*

RFCs	Title
RFC 2082	*RIP-2 MD5 Authentication*

IETF RFCs