

Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide, Release 10.6(x)

First Published: 2025-08-13

Last Modified: 2026-04-27

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

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS REFERENCED IN THIS DOCUMENTATION ARE SUBJECT TO CHANGE WITHOUT NOTICE. EXCEPT AS MAY OTHERWISE BE AGREED BY CISCO IN WRITING, ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS DOCUMENTATION ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED.

The Cisco End User License Agreement and any supplemental license terms govern your use of any Cisco software, including this product documentation, and are located at: https://www.cisco.com/c/en/us/about/legal/cloud-and-software/software-terms.html. Cisco product warranty information is available at https://www.cisco.com/c/en/us/products/warranty-listing.html. US Federal Communications Commission Notices are found here https://www.cisco.com/c/en/us/products/us-fcc-notice.html.

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 products and features described herein as in development or available at a future date remain in varying stages of development and will be offered on a when-and if-available basis. Any such product or feature roadmaps are subject to change at the sole discretion of Cisco and Cisco will have no liability for delay in the delivery or failure to deliver any products or feature roadmap items that may be set forth in this document.

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.

The documentation set for this product strives to use bias-free language. For the purposes of this documentation set, bias-free is defined as language that does not imply discrimination based on age, disability, gender, racial identity, ethnic identity, sexual orientation, socioeconomic status, and intersectionality. Exceptions may be present in the documentation due to language that is hardcoded in the user interfaces of the product software, language used based on RFP documentation, or language that is used by a referenced third-party product.

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)

© 2025–2026 Cisco Systems, Inc. All rights reserved.



CONTENTS

Trademarks ?

PREFACE

Preface xxxi

Audience xxxi

Document Conventions xxxi

Related Documentation for Cisco Nexus 9000 Series Switches xxxii

Documentation Feedback xxxii

Communications, services, and additional information xxxii

Cisco Bug Search Tool xxxiii

Documentation feedback xxxiii

CHAPTER 1

New and changed information 1

New and changed information 1

CHAPTER 2

Overview 7

Licensing Requirements 7

Supported Platforms 7

Layer 3 Unicast Routing 7

Routing Fundamentals 8

Packet Switching 8

Route Metrics 9

Path Lengths 9

Reliability 10

Routing Delays 10

Bandwidth 10

Load 10

Communication Cost 10
Router IDs 10
Autonomous Systems 11
Convergence 12
Load Balancing and Equal Cost Multipath 12
Route Redistribution 12
Administrative Distances 12
Stub Routing 13
Routing Algorithms 14
Static Routes and Dynamic Routing Protocols 14
Interior and Exterior Gateway Protocols 14
Distance Vector Protocols 14
Link-State Protocols 15
Layer 3 Virtualization 16
Cisco NX-OS Forwarding Architecture 16
Unicast RIB 16
Adjacency Manager 17
Unicast Forwarding Distribution Module 17
FIB 17
Hardware Forwarding 18
Software Forwarding 18
Guidelines and Limitations on the Cisco N9364E-SG2-Q Switch 18
Summary of Layer 3 Unicast Routing Features 18
IPv4 and IPv6 Addresses 19
IP Services 19
OSPF 19
EIGRP 19
IS-IS 19
BGP 20
RIP 20
Static Routing 20
Layer 3 Virtualization 20
Route Policy Manager 20
Policy-Based Routing 20

First Hop Redundancy Protocols 21
Object Tracking 21
Related Topics 21

CHAPTER 3 IPv4 Addresses 23
IPv4 Addresses 23
Multiple IPv4 Addresses 24
LPM routing modes 24
Host to LPM Spillover 26
Address Resolution Protocol 27
ARP cache 27
Static and dynamic entries in the ARP Cache 28
Devices that do not use ARP 28
Reverse ARP 28
Proxy ARP 29
Local proxy ARP 30
Gratuitous ARP 30
Periodic ARP refresh on MAC delete 30
ARP refresh on MAC Delete 30
Glean throttles 30
Path MTU discovery 31
ICMP 31
Virtualization support for IPv4 31
Prerequisites for IPv4 31
Guidelines and Limitations for IPv4 31
Default settings 34
Configure IPv4 35
Configure IPv4 address 35
Configure multiple IP address 36
Configure max-host routing mode 37
Configure nonhierarchical routing mode (Cisco Nexus 9500 platform switches only) 38
Configure 64-Bit ALPM routing mode (Cisco Nexus 9500 platform switches only) 39
Configure ALPM routing mode (Cisco Nexus 9300 platform switches only) 40

Configure LPM heavy routing mode (Cisco Nexus 9200 and 9300-EX platform switches and 9732C-EX line card only) 41
Configuring LPM Internet-Peering Routing Mode 42
Configure LPM dual-host routing mode 43
Configure a Static ARP Entry 44
Configure proxy ARP 44
Configure local proxy ARP on ethernet interfaces 45
Configuring local proxy ARP on SVIs 46
Configuring periodic ARP refresh on MAC delete for SVIs 47
Configure gratuitous ARP 48
Configure out of subnet ARP resolution 48
Configure ARP cache limit per SVI interface 49
Configure path MTU discovery 50
Configure IP directed broadcasts 50
Configuring IP glean throttling 51
Configure the hardware IP glean throttle maximum 52
Configure the hardware IP glean throttle timeout 52
Configure the interface IP address for the ICMP source IP field 53
Configure IPv4 redirect syslog 54
Verify the IPv4 Configuration 54
Additional References 55
Related Documents for IPv4 55

CHAPTER 4 **IPv6 Addresses 57**
IPv6 addresses 57
IPv6 Address Formats 58
IPv6 unicast addresses 59
Aggregatable global addresses 59
Link-Local addresses 60
IPv4-Compatible IPv6 addresses 61
Unique local addresses 61
Site local addresses 62
IPv4 packet header 62
Simplified IPv6 packet headers 62

DNS for IPv6 66
Path MTU discovery for IPv6 66
CDP IPv6 address support 67
ICMP for IPv6 67
IPv6 neighbor discovery 68
IPv6 neighbor solicitation messages 68
IPv6 stateless autoconfigurations 70
IPv6 compute node IP autoconfigurations 71
IPv6 router advertisement messages 71
IPv6 neighbor redirect messages 73
IPv6 anycast addresses 74
IPv6 multicast addresses 74
LPM routing modes 75
Host to LPM spillover 78
Virtualization support 78
IPv6 routes with ECMP 79
Prerequisites for IPv6 79
Guidelines and limitations for IPv6 79
Configure IPv6 82
Configure IPv6 addresses 82
Configure max-host routing mode (Cisco Nexus 9500 Platform switches only) 83
Configure nonhierarchical routing mode (Cisco Nexus 9500 series switches only) 84
Configure 64-Bit ALPM routing mode (Cisco Nexus 9500 platform switches only) 85
Configure ALPM routing mode (Cisco Nexus 9300 platform switches only) 87
Configure IPv6 neighbor discovery 87
Optional IPv6 Neighbor Discovery 89
Configure IPv6 ND cache limit at the SVI 90
Configure IPv6 packet verification 91
Configure IPv6 stateless autoconfiguration 93
Configure LPM heavy routing mode (Cisco Nexus 9200 and 9300-EX platform switches and 9732C-EX Line card only) 94
Configure L3 host heavy template 95
Configure LPM internet-peering routing mode (Cisco Nexus 9500-R platform switches, Cisco Nexus 9300-EX platform switches and Cisco Nexus 9000 series switches with 9700-EX line cards only) 96

Additional configuration for LPM internet-peering routing mode 97
Configure LPM dual-host routing mode (Cisco Nexus 9200 and 9300-EX platform switches) 98
Configure IPv6 redirect syslog 99
Verify the IPv6 configuration 100
Verify the IPv6 configuration 100

CHAPTER 5 DNS 103
DNS clients 103
DNS client 103
Name Servers 103
DNS operations 104
High availability 104
Virtualization support 104
Prerequisites for DNS clients 104
Guidelines and limitations for DNS clients 104
Default settings for DNS clients 105
Configure DNS clients 105
Configure DNS client 105
Configure virtualization 107
Verify DNS client configuration 109
Configuration examples for the DNS client 109

CHAPTER 6 OSPFv2 111
OSPFv2 111
OSPFv2 and the unicast RIB 112
Authentication 112
Simple password authentication 113
Cryptographic authentication 113
MD5 authentication 113
HMAC-SHA authentication 113
Advanced features 113
Stub areas 113
Not so stubby areas 114
Virtual links 115

Route redistribution 115
Route summarization 116
High availability and graceful restart 116
OSPFv2 stub router advertisements 117
Multiple OSPFv2 instances 117
SPF Optimization 117
BFD 118
Virtualization support for OSPFv2 118
Prerequisites for OSPFv2 118
Guidelines and limitations for OSPFv2 118
Default settings for OSPFv2 120
Configure Basic OSPFv2 121
Enable OSPFv2 121
Create an OSPFv2 instance 122
Configure Optional Parameters on an OSPFv2 Instance 123
Configure networks in OSPFv2 124
Configure authentication for an area 126
Configure authentication for an interface 128
Configure Advanced OSPFv2 131
Configure filter lists for border routers 131
Configure stub areas 132
Configure a totally stubby area 133
Configure NSSA 133
Configure multi-area adjacency 136
Configure virtual links 137
Configure redistribution 139
Limit the number of redistributed routes 141
Configure route summarization 143
Configure stub route advertisements 144
Configure the administrative distance of routes 145
Modify the default timers 147
Configure graceful restart 150
Restart an OSPFv2 instance 151
Configure OSPFv2 with virtualization 152

Verify the OSPFv2 configuration 153
Monitor OSPFv2 155
Configuration examples for OSPFv2 155
OSPF RFC compatibility mode example 155
Related Documents for OSPFv2 156
Related Documents for OSPFv2 156
MIBs 156

CHAPTER 7 OSPFv3 157
OSPFv3 157
Comparison of OSPFv3 and OSPFv2 158
Hello packets 158
Neighbors 159
Adjacency 159
Designated routers 160
Areas 160
Link-State advertisements 161
Link-State advertisement types 161
Link cost 162
Flooding and LSA group pacing 162
Link-State database 163
Multi-Area adjacency 163
OSPFv3 and the IPv6 unicast RIB 163
Address family support 164
Authentication and encryption 164
Advanced features 164
Stub areas 164
Not-So-Stubby Area 165
Virtual links 166
Route redistribution 166
Route summarization 167
High Availability and Graceful Restart 167
Multiple OSPFv3 instances 168
SPF Optimization 168

BFD 169
Virtualization support 169
Prerequisites for OSPFv3 169
Guidelines and limitations for OSPFv3 169
Default settings 171
Configure basic OSPFv3 172
Enable OSPFv3 172
Create an OSPFv3 instance 173
Configure networks in OSPFv3 175
Configure advanced OSPFv3 177
Configure filter lists for border routers 177
Configure stub areas 179
Configure a totally stubby area 180
Configure NSSA 180
Configure multi-area adjacency 182
Configure virtual links 183
Configure redistribution 185
Limit the number of redistributed routes 187
Configure route summarization 189
Configure the administrative distance of routes 190
Modify the default timers 193
Configure graceful restart 195
Restart an OSPFv3 instance 197
Configure OSPFv3 with virtualization 197
Configure Encryption and Authentication 199
Configuring OSPFv3 encryption at router level 200
Configure OSPFv3 encryption at area level 201
Configure OSPFv3 encryption at interface level 202
Configure OSPFv3 encryption for virtual links 203
Configure OSPFv3 authentication at router level 205
Configure OSPFv3 authentication at area level 206
Configure OSPFv3 authentication at interface level 208
Configure OSPFv3 authentication at virtual links level 209
Verify the OSPFv3 configuration 210

Monitor OSPFv3 211
Configuration examples for OSPFv3 212
Related topics 212
Additional references 212
MIBs 213

CHAPTER 8 EIGRP 215
EIGRP 215
EIGRP Components 215
Reliable transport protocol 216
Neighbor discovery and recovery 216
Diffusing update algorithm 216
EIGRP route updates 217
Internal route metrics 217
Wide metrics 218
External route metrics 218
EIGRP and the unicast RIB 219
Advanced EIGRP 219
Address families 219
Authentication 219
Stub routers 220
Route summarization 220
Route redistribution 220
Load balancing 221
Split horizon 221
BFD 221
Virtualization support 222
Graceful restart and high availability 222
Multiple EIGRP instances 222
Prerequisites for EIGRP 222
Guidelines and limitations for EIGRP 223
Default settings 224
Configuring Basic EIGRP 225
Enable the EIGRP Feature 225

Create an EIGRP Instance 226
Restart an EIGRP Instance 229
Shutting Down an EIGRP Instance 229
Configure a Passive Interface for EIGRP 229
Shut Down EIGRP on an Interface 230
Configure Advanced EIGRP 230
Configure Authentication in EIGRP 230
Configuring EIGRP Stub Routing 232
Configure a Summary Address for EIGRP 234
Redistribute Routes into EIGRP 234
Limit the Number of Redistributed Routes 236
Configure Load Balancing in EIGRP 238
Configure Graceful Restart for EIGRP 239
Adjust the Interval Between Hello Packets and the Hold Time 241
Disable Split Horizon 242
Enable Wide Metrics 242
Tune EIGRP 242
Configure Virtualization for EIGRP 245
Verify the EIGRP Configuration 247
Monitor EIGRP 247
Configuration Examples for EIGRP 248
Related Topics 249
Additional References 249
Related Documents 249
MIBs 249

CHAPTER 9 **Configure IS-IS 251**
IS-IS 251
IS-IS hello packets and adjacency formation 252
IS-IS areas 252
NET and system ID 253
Designated intermediate systems 253
IS-IS authentication 254
Mesh groups 254

Overload bits 254
Route summarization 255
Route redistribution 255
Link prefix suppression 255
Load balance 255
BFD 256
Virtualization support 256
High availability and graceful restart 256
Multiple IS-IS instances 257
Prerequisites for IS-IS 257
Guidelines and limitations for IS-IS 257
Default settings 257
Configure IS-IS 258
IS-IS Configuration Modes 258
Enable the IS-IS Feature 259
Create an IS-IS Instance 259
Restart an IS-IS Instance 262
Shut Down IS-IS 262
Configure IS-IS on an Interface 262
Shut Down IS-IS on an Interface 264
Configure IS-IS Authentication in an Area 264
Configure IS-IS Authentication on an Interface 266
Configure a Mesh Group 267
Configure a Designated Intermediate System 267
Configuring Dynamic Host Exchange 268
Set the Overload Bit 268
Configure the Attached Bit 269
Configure the Transient Mode for Hello Padding 269
Configure a Summary Address 269
Configuring Redistribution 271
Limiting the number of redistributed routes 272
Advertise Only Passive Interface Prefixes 274
Suppress Prefixes on an Interface 275
Disable Strict Adjacency Mode 276

Configure a Graceful Restart 278
Configure Virtualization 279
Tune IS-IS 281
Verifying the IS-IS Configuration 283
Monitor IS-IS 284
Configuration Examples for IS-IS 285
Related Topics 285

CHAPTER 10 BGP 287

BGP 287
BGP 287
BGP autonomous systems 288
4-Byte AS number support 288
Administrative distance 288
BGP peers 288
BGP sessions 289
Dynamic AS numbers for prefix peers and interface peers 289
BGP router identifier 289
BGP path selection 290
BGP path selection - comparing pairs of paths 291
BGP path selection - determining the order of comparisons 292
BGP path selection - determining the best-path change suppression 293
BGP and the unicast RIB 293
BGP prefix independent convergence 293
BGP PIC edge unipath 294
BGP PIC edge with multipath 296
BGP PIC core 298
BGP PIC feature support matrix 299
BGP virtualization 299
Prerequisites 299
Guidelines and limitations for basic and advanced BGP 299
Default Settings 305
Configure Basic BGP 305
CLI Configuration Modes 305
Global Configuration Mode 305

Address Family Configuration Mode 306
Neighbor Configuration Mode 306
Neighbor Address Family Configuration Mode 306
Configure Basic BGP 307
Enable BGP 307
Create a BGP Instance 308
Restart a BGP Instance 310
Shut Down BGP 310
Configure BGP Peers 311
Configure Dynamic AS Numbers for Prefix Peers 313
Configure BGP PIC Edge 315
Configure BGP PIC Core 317
Clear BGP Information 318
Verify the Basic BGP Configuration 321
Monitor BGP Statistics 323
Configuration Examples for Basic BGP 324
Related topics 324
Where to go next 324
Additional References 324
Where to Go Next 324
Configure advance BGP 324
Advanced BGP 324
Peer templates 325
Authentication 325
Route policies and resetting BGP sessions 325
eBGP 326
iBGP 327
Capabilities negotiation 329
Route dampening 329
Load sharing and multipath 329
BGP additional paths 330
Route aggregation 331
BGP conditional advertisement 332
BGP next-hop address tracking 332

Route redistribution 333
Labeled and unlabeled unicast routes 333
BFD 333
Tune BGP 333
Multiprotocol BGP 334
BGP monitoring protocol 336
Graceful restart and high availability 336
Low memory handling 337
Virtualization support 338
Enable IP forward on an interface 338
Configure BGP Session Templates 338
Configure BGP Peer-Policy Templates 341
Configure BGP Peer Templates 343
Configure Prefix Peering 346
Configure BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families 347
Configure BGP Authentication 351
Configure TCP authentication option 352
About TCP authentication option 353
TCP-AO key chain 353
TCP-AO key rollover 355
Guidelines and limitations 355
Configure TCP key chain and keys 356
TCP keychain verification commands 358
Configuration example for a TCP keychain 359
Resource Public Key Infrastructure (RPKI) 359
RPKI Configuration 360
Commands for connecting to RPKI caches 360
Commands for marking incoming prefixes with RPKI validation state 361
Commands for using RPKI validation state in BGP best-path-computation 362
Commands for dropping out or manipulating prefixes with specific validation states using route-map 362
RPKI Show Commands 363
RPKI Clear Commands 364
RPKI Debug and Event History Commands 364

Resetting a BGP Session 365
Modify the Next-Hop Address 366
Configure BGP Next-Hop Address Tracking 366
Configure BGP Next-Hop Address Tracking 367
Configuring Next-Hop Resolution via Default Route 367
Control Reflected Routes Through Next-Hop-Self 368
Shrink Next-Hop Groups When A Session Goes Down 368
Disable Capabilities Negotiation 369
Disable Policy Batching 369
Configure BGP Additional Paths 370
Advertise the Capability of Sending and Receiving Additional Paths 370
Configure the Sending and Receiving of Additional Paths 371
Configure Advertised Paths 372
Configure Additional Path Selection 373
Configuring eBGP 374
Disable eBGP Single-Hop Checking 374
Configure TTL Security Hops 374
Configure eBGP Multihop 376
Disable a Fast External Fallover 377
Limit the AS-path Attribute 377
Configure Local AS Support 378
Configure AS Confederations 378
Configure Route Reflector 379
Configure Next-Hops on Reflected Routes Using an Outbound Route-Map 381
Configure Route Dampening 384
Configure Load Sharing and ECMP 384
Unequal Cost Multipath (UCMP) over BGP 384
Enable UCMP over BGP 385
Guidelines and Limitations for UCMP over BGP 385
Configure Maximum Prefixes 385
Configure DSCP 386
Configure Dynamic Capability 386
Configure Aggregate Addresses 387
Suppress BGP Routes 388

Configure BGP Conditional Advertisement 388
Configure Route Redistribution 390
DMZ Link Bandwidth 392
Guidelines and Limitations 393
Configure BGP DMZ Link Bandwidth 393
Configuration Examples for BGP DMZ Link Bandwidth 395
Configure Unequal Cost Multipath (UCMP) Using Link Bandwidth Extended Community 397
Configuration Example 398
Verify Configuration 403
Advertise the Default Route 404
Configure BGP Attribute Filtering and Error Handling 405
Treat as Withdraw Path Attributes from a BGP Update Message 405
Discarding Path Attributes from a BGP Update Message 406
Enable or Disable Enhanced Attribute Error Handling 406
Display Discarded or Unknown Path Attributes 407
Tune BGP 408
Configure Policy-Based Administrative Distance 412
Configure Multiprotocol BGP 414
Configure BMP 415
BGP Local Route Leaking 418
BGP Local Route Leaking 418
Guidelines and Limitations for BGP Local Route Leaking 418
Configure Routes Imported from a VPN to Leak into the Default VRF 418
Configure Routes Leaked from the Default-VRF to Export to a VPN 419
Configure Routes Imported from a VPN to Export to a VRF 420
Configure Routes Imported from a VRF to Export to a VPN 421
Configuration Examples 422
Display BGP Local Route Leaking Information 425
BGP Graceful Shutdown 425
BGP Graceful Shutdown 425
Graceful Shutdown Aware and Activate 425
Graceful Shutdown Contexts 426
Graceful Shutdown with Route Maps 427
Guidelines and Limitations 428

Graceful Shutdown Task Overview 428
Configure Graceful Shutdown on a Link 429
Filter BGP Routes and Setting Local Preference Based On GRACEFUL_SHUTDOWN Communities 430
Configure Graceful Shutdown for All BGP Neighbors 432
Control the Preference for All Routes with the GRACEFUL_SHUTDOWN Community 432
Prevent Sending the GRACEFUL_SHUTDOWN Community to a Peer 433
Display Graceful Shutdown Information 434
Graceful Shutdown Configuration Examples 435
Configure a Graceful Restart 437
Configure Virtualization 439
Verify the Advanced BGP Configuration 441
Monitor BGP Statistics 443
Configuration Examples 443
Related Topics 444
Additional References 444
MIBs 445

CHAPTER 11 **Configure RIP** 447
RIP 447
RIP 447
RIPv2 Authentication 447
Split Horizons 448
Route Filtering 448
Route Summarizations 448
Route Redistributions 449
Load Balancing 449
High Availability for RIP 449
Virtualization Support for RIP 449
Prerequisites for RIP 449
Guidelines and Limitations for RIP 449
Default Settings for RIP Parameters 450
RIPs 450
Enable RIP 450

Create a RIP Instance 451
Restart the RIP Instance 452
Configure RIP on an Interface 453
Configure RIP Authentication 453
Configure a Passive Interface 455
Configure Split Horizon With Poison Reverse 455
Configure Route Summarization 455
Configure Route Redistribution 456
Configure Cisco NX-OS RIP for Compatibility with Cisco IOS RIP 457
Configure Virtualization 459
Tune the RIP Protocol 461
Verify the RIP Configuration 462
Display the RIP Statistics 463
Configuration Examples for RIP 463
Related Topics 464

CHAPTER 12 **Configure RIPng** 465
RIPng 465
RIPng 465
Split Horizon 465
Route filter 466
Load balance 466
Default information origination and generation 466
High availability for RIPng 467
Virtualization for RIPng 467
Prerequisites for RIPng 467
Guidelines and limitations for RIPng 467
Default settings for RIPng parameters 468
Configure RIPng 468
Enable RIPng 468
Create an RIPng instance 469
Restart an RIPng instance 470
Configure RIPng on an interface 471
Configure split horizon with poison reverse 472

Configure Cisco NX-OS RIPng for compatibility with Cisco IOS RIPng 472
Configure virtualization 474
Tune RIPng 476
Verify the RIPng configuration 477
Display RIPng statistics 478
Configuration Examples for RIPng 478
Related topics 478

CHAPTER 13 Configure static route 479
About static route 479
Administrative distance 479
Directly connected static routes 480
Fully specified static routes 480
Floating static routes 480
Remote next hops for static routes 480
BFD 480
Virtualization support 481
Prerequisites for static routing 481
Default settings 481
Configuring static routing 481
Configure a static route 481
Configure a static route over a VLAN 483
Configure virtualization 484
Verify the static routing configuration 486
Configuration example for static route 486

CHAPTER 14 Configure Layer 3 virtualization 487
Layer 3 virtualization 487
VRF and routing 488
Route leaking and importing routes from the default VRF 489
BGP VRF router-ID for IPv6 only environments 489
VRF-Aware services 490
Reachability 491
Filtering 491

Combine reachability and filtering 491
Prerequisites for VRF 492
Guidelines and limitations for VRFs 492
Guidelines and limitations for VRF route leaking 493
Default settings 493
Configure VRFs 494
Create VRF 494
Assign VRF membership to an interface 495
Configure VRF parameters for a routing protocol 496
Configure a VRF-Aware service 498
Set the VRF scope 499
Verify the VRF configuration 500
Configuration examples for VRFs 500
Additional references 508
Related documents for VRFs 508
Standards 508

CHAPTER 15 **Manage the Unicast RIB and FIB** 509
About the unicast RIB and FIB 509
Layer 3 consistency checker 510
Guidelines and limitations for the unicast RIB 510
Manage the unicast RIB and FIB 511
Display the module FIB information 511
Configure load sharing in the unicast FIB 512
Dynamic load balancing 514
Display the routing and adjacency information 514
Trigger the layer 3 consistency checker 516
Clear the FIB forwarding information 517
Configure maximum routes for the unicast RIB 517
Estimate memory requirements for routes 518
Clear routes in the unicast RIB 519
Verify the unicast RIB and FIB configuration 520
Additional references 521
Related documents 521

CHAPTER 16 **Configuring Route Policy Manager** 523

About route policy manager 523

Prefix lists 523

MAC lists 524

Route maps 524

Default action for sequences in a route map 525

Default sequence number for a route map 525

Match criteria 525

Set changes 526

Access lists 526

AS numbers for BGP 526

AS-path lists for BGP 527

Community lists for BGP 527

Extended community lists for BGP 527

Configure NX-OS BGP large communities 528

Route redistribution and route maps 533

Route map support matrix for routing protocols 534

Guidelines and limitations for route policy manager 537

Default settings for route policy manager parameters 538

Configure route policy manager 539

Configure IP prefix lists 539

Configure MAC lists 541

Configure AS-path lists 543

Replace BGP AS-path attribute 544

Replace the complete AS-path 544

Replace selected AS numbers in the AS-path 546

Configure community lists 547

Configure extended community lists 548

Configure route maps 550

Global commands to block the deletion of route-map 552

Verify the route policy manager configuration 553

Configuration examples for route policy manager 553

Related topics 553

CHAPTER 17 **Configuring Policy-Based Routing** 555

About policy-based routing 555

Policy route maps 556

Set criteria for policy-based routing 556

Route map support matrix for policy-based routing 557

Route-map processing logic 558

Prerequisites for policy-based routing 559

Guidelines and limitations for policy-based routing 559

Default settings for policy-based routing 563

Configure policy-based routing 563

Enable the policy-based routing feature 563

Enable the policy-based routing over ECMP 564

Configure PBR fast convergence 565

Configure a route policy 566

Redirect to VRF Using a Route Map 571

Configure Set VRF 572

Redirect default route match to next-hop 574

Verify the policy-based routing configuration 576

Configuration examples for policy-based routing 576

Related documents for policy-based routing 579

CHAPTER 18 **Configure Dynamic Load Balancing** 581

Dynamic Load Balancing mechanism 581

Features 582

How DLB topology works in AI/ML networks 582

Dynamic Load Balancing feature in NX-OS releases 584

Dynamic Load Balancing on Nexus switches 584

Dynamic Load Balancing on CloudScale switches 584

Key concepts of Dynamic Load Balancing on CloudScale switches 584

Guidelines and Limitations for Dynamic Load Balancing on CloudScale switches 586

Configure Dynamic Load Balancing on CloudScale switches 588

Calculate Link Utilization Level 591

Dynamic Load Balancing on Silicon One switches 593

Key concepts of Dynamic Load Balancing on Silicon One switches 593
Guidelines and limitations for Dynamic Load Balancing on Silicon One switches 597
Configure Dynamic Load Balancing on Silicon One switches 599
Dynamic Load Balancing on NVIDIA Spectrum switches 603
Key concepts of Dynamic Load Balancing on NVIDIA Spectrum switches 604
Guidelines and limitations for Dynamic Load Balancing on NVIDIA Spectrum switches 604
Configure Dynamic Load Balancing on NVIDIA Spectrum switches 605
Configuration samples for Dynamic Load Balancing 605
Verify the Dynamic Load Balancing configuration 606
Troubleshoot Dynamic Load Balancing 608

CHAPTER 19 **Configuring HSRP** 609
About HSRP 609
HSRP overview 610
HSRP versions 611
HSRP for IPv4 612
HSRP for IPv6 612
HSRP for IPv6 addresses 613
Multiple group optimization for HSRP 614
HSRP subnet VIP 614
HSRP authentication 614
HSRP messages 614
HSRP load sharing 615
Object tracking and HSRP 616
vPCs and HSRP 617
vPC peer gateway and HSRP 617
BFD 618
High availability and extended nonstop forwarding 618
Virtualization support 619
Prerequisites for HSRP 619
Guidelines and limitations for HSRP 619
Default settings for HSRP parameters 621
Configue HSRP 622
Enable HSRP 622

Configure the HSRP version 622
Configure an HSRP group for IPv4 622
Configure an HSRP group for IPv6 624
Configure an HSRP master group task 626
Configure an HSRP slave group 628
Configure the HSRP virtual MAC address 630
Authenticate HSRP 631
Configure HSRP object tracking 632
Configure the HSRP priority 634
Customize HSRP in HSRP configuration mode 635
Customize HSRP in interface configuration mode 636
Configure extended hold timers for HSRP 637
Verify the HSRP Configuration 638
Configuration examples for HSRP 639
Additional references 640
Related documents 640
MIBs 641

CHAPTER 20 **Configure VRRP** 643
About VRRP 643
VRRP operation 643
VRRP benefits 645
Multiple VRRP groups 646
VRRP router priority and preemption 647
vPCs and VRRP 648
VRRP advertisements 649
VRRP authentication 649
VRRP tracking 650
BFD for VRRP 650
Information about VRRPv3 and VRRS 651
VRRPv3 benefits 651
VRRPv3 object tracking 651
High availability 652
Virtualization support 652

Guidelines and limitations for VRRP 652
Guidelines and limitations for VRRPv3 653
Default settings for VRRP parameters 653
Default settings for VRRPv3 parameters 654
Configure VRRP 654
Enable VRRP 654
Configure a VRRP group 655
Configure VRRP priority 656
Configure VRRP authentication 658
Configure time intervals for advertisement packets 659
Disable preemption 661
Configure VRRP interface state tracking 662
Configure VRRP object tracking 663
Configure VRRPv3 665
Enable VRRPv3 and VRRS 665
Create VRRPv3 groups 665
Configure VRRPv3 control groups 668
Configure VRRPv3 object tracking 669
Configure VRRS pathways 670
Verify the VRRP configuration 672
Verify the VRRPv3 configuration 672
Monitor and clear VRRP statistics 673
Monitor and clear VRRPv3 statistics 673
Configuration examples for VRRP 673
Configuration examples for VRRPv3 674
Additional references 676
Related documents for VRRP 676

CHAPTER 21 **Configuring Object Tracking** 677
Information about object tracking 677
Object tracking overview 677
Object track list 678
High availability 679
Virtualization support 679

Configuration examples for object tracking 679
Guidelines and limitations for object tracking 680
Default settings 680
Configure object tracking 681
Configure object tracking for an interface 681
Delete a tracking object 682
Configure object tracking for route reachability 682
Configure an object track list with a boolean expression 683
Configure an object track list with a percentage threshold 685
Configure an object track list with a weight threshold 686
Configure an object tracking delay 687
Configure object tracking for a nondefault VRF 689
Verify the object tracking configuration 691
Configuration examples for object tracking 691
Related topics 692
Related documents 692

APPENDIX A **IETF RFCs supported by Cisco NX-OS unicast features** 693
BGP RFCs 693
First-hop redundancy protocols RFCs 694
IP services RFCs 695
IPv6 RFCs 695
IS-IS RFCs 696
OSPF RFCs 697
RIP RFCs 697

APPENDIX B **Configuration Limits for Cisco NX-OS Layer 3 Unicast Features** 699
Configuration limits for Cisco NX-OS layer 3 unicast features 699
Layer 3 heavy routing template 699
Guidelines and limitations for layer 3 heavy routing template 700
Layer 3 heavy routing template scale limits 700
Configure the layer 3 heavy routing template 701
Verify layer 3 heavy routing template 702



Preface

This preface includes the following sections:

- Audience, on page xxxi
- Document Conventions, on page xxxi
- Related Documentation for Cisco Nexus 9000 Series Switches, on page xxxii
- Documentation Feedback, on page xxxii
- Communications, services, and additional information, on page xxxii

Audience

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

Document Conventions

Command descriptions use the following conventions:

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

Convention	Description
`variable`	Indicates a variable for which you supply values, in context where italics cannot be used.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string includes the quotation marks.

Examples use the following conventions:

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

Related Documentation for Cisco Nexus 9000 Series Switches

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

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

Documentation Feedback

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

Communications, services, and additional information

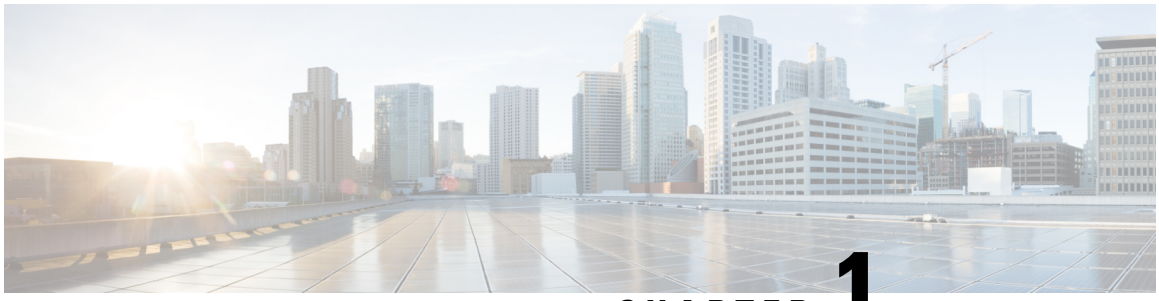
- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business impact you're looking for with the technologies that matter, visit Cisco Services.
- To submit a service request, visit Cisco Support.
- To discover and browse secure, validated enterprise-class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

Cisco Bug Search Tool

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

Documentation feedback

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



CHAPTER 1

New and changed information

- New and changed information, on page 1

New and changed information

This table summarizes the new and changed features for the *Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide, Release 10.6(x)*.

Table 1: New and Changed Features

Feature	Description	Changed in Release	Where Documented
Undefined Policy "Permit by Default"	Administrators can change the default action to "permit," ensuring traffic continuity while configurations are being applied or updated.	10.6(3)F	Route policies and resetting BGP sessions
New HW templates for N9K-C9364C-H1, N9K-C9332D-H2R, and N9K-C93400LD-H1	Added support for **template-l3-host-heavy** on N9K-C9364C-H1, N9K-C9332D-H2R, and N9K-C93400LD-H1.	10.6(3)F	Guidelines and limitations for IPv6, on page 79 Configure L3 host heavy template, on page 95
IPv6 ND cache limit at the SVI	Added support to configure IPv6 ND cache limit at the SVI.	10.6(3)F	Guidelines and limitations for IPv6, on page 79 Configure IPv6 ND cache limit at the SVI , on page 90
Proxy ARP enhancement	Enhaces Proxy ARP capabilities to improve security, scalability, and operational flexibility in environments with Layer-2 isolation, VXLAN, and EVPN deployments.	10.6(3)F	Configuring local proxy ARP on SVIs, on page 46

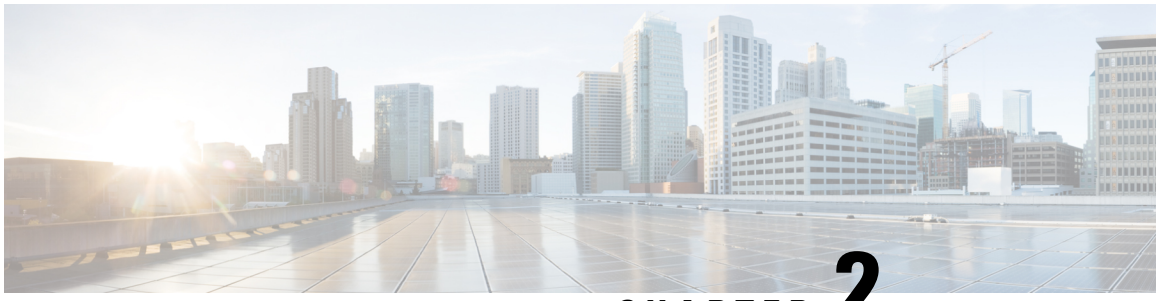
Feature	Description	Changed in Release	Where Documented
Layer 3 ECMP Dynamic Load Balancing (DLB)	Beginning with Cisco NX-OS Release 10.6(2n), the Cisco N9164E-NS4-O switch supports Layer 3 ECMP Dynamic Load Balancing (DLB) .	10.6(2n)	Dynamic Load Balancing feature in NX-OS releases
L3 Support (IPv4, IPv6)	Beginning with Cisco NX-OS Release 10.6(2n), Layer 3 IPv4 and IPv6 support is available on the Cisco N9164E-NS4-O switch.	10.6(2n)	Guidelines and limitations for IPv6 Guidelines and Limitations for IPv4
DLB on BGP Unnumbered Link Local IPv6 with WCMP.	Beginning with Cisco NX-OS Release 10.6(2)F, you can configure DLB on BGP Unnumbered Link Local IPv6 with WCMP.	10.6(2)F	Configure BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families
Layer 3 Heavy Routing Template for CloudScale switches	Added support for the Layer 3 Heavy Routing Template, which reallocates hardware resources to maximize IPv4 and IPv6 route scale.	10.6(2)F	Layer 3 heavy routing template, on page 699 Guidelines and limitations for layer 3 heavy routing template, on page 700 Layer 3 heavy routing template scale limits, on page 700 Configure the layer 3 heavy routing template, on page 701 Verify layer 3 heavy routing template, on page 702
Namespace-specific DNS resolution for third-party Linux applications in containers	Beginning with Cisco NX-OS Release 10.6(2)F, Cisco Nexus 9000 Series switches support namespace-specific DNS resolution for third-party Linux applications running in containers.	10.6(2)F	Guidelines and Limitations for DNS Clients

Feature	Description	Changed in Release	Where Documented
Unicast Routing/ L3 (IPv4 and IPv6)	Added support for the following features on N9324C-SE1U and N9348Y2C6D-SE1U: • OSPF • EIGRP • ISIS • RIP • VRF • ECMP, 8192 ECMP groups, 512-way ECMP • IP directed broadcast • uRPF • IPv6 • Static routing, PBR • IP unnumbered (non-SVI) • HSRP, VRRP	10.6(2)F	Guidelines and Limitations for RIP, on page 449 Guidelines and limitations for VRFs, on page 492 Guidelines and limitations for EIGRP, on page 223 Guidelines and limitations for OSPFv3, on page 169 Guidelines and limitations for OSPFv2, on page 118 Guidelines and limitations for basic and advanced BGP, on page 299 Configuring static routing, on page 481 Guidelines and limitations for HSRP, on page 619 Guidelines and limitations for VRRPv3, on page 653 Guidelines and Limitations for IPv4, on page 31 Guidelines and Limitations for PIM and PIM6
Policy-driven DLB on CloudScale switches	Added support for Policy-driven DLB mode on CloudScale switches.	10.6(2)F	Dynamic Load Balancing on CloudScale switches, on page 584
DLB Static pinning support on N9364E-SG2 switches	Added support for DLB static pinning on Cisco N9364E-SG2-Q and N9364E-SG2-O switches.	10.6(2)F	Dynamic Load Balancing on Silicon One switches, on page 593 Configuration samples for Dynamic Load Balancing, on page 605 Verify the Dynamic Load Balancing configuration, on page 606
New HW template to handle /128 host route	Added support for template-l3-host-heavy to handle /128 host routes.	10.6(2)F	Guidelines and limitations for IPv6, on page 79 Configure L3 host heavy template, on page 95

Feature	Description	Changed in Release	Where Documented
Hardware assisted IP directed broadcast	Added support for hardware assisted IP directed broadcast.	10.6(2)F	Guidelines and Limitations for IPv4, on page 31 Configure IP directed broadcasts, on page 50
NX-OS BGP Maximum-prefix [discard-extra-paths] feature support	Added support for NX-OS BGP Maximum-prefix [discard-extra-paths] feature.	10.6(2)F	Guidelines and limitations for basic and advanced BGP, on page 299 Configure Maximum Prefixes, on page 385
Support for large BGP Communities over EVPN address family	Added support for BGP Large Communities over EVPN address family.	10.6(2)F	Guidelines and limitations for route policy manager, on page 537 About NX-OS BGP large communities, on page 528
AI/ML License Enhancements (Adaptive routing)	Added support to utilize the end-to-end Adaptive Routing capability between qualified Cisco Nexus switches and NVIDIA® Spectrum-X SmartNICs.	10.6(1)F	Configure Dynamic Load Balancing, on page 581
Support BGP Prefix Origin Validation State Extended Community when connected via eBGP.	Added support for signaling of BGP Prefix Origin Validation State Extended Community towards eBGP peers.	10.6(1)F	Resource Public Key Infrastructure (RPKI), on page 359 Commands for marking incoming prefixes with RPKI validation state, on page 361
Support for Breakout interfaces with DLB	Added support for breakout ports on N9364E-SG2 switches for the Dynamic Load Balancing feature.	10.6(1)F	Guidelines and limitations for Dynamic Load Balancing on Silicon One switches, on page 597 Configure Dynamic Load Balancing on Silicon One switches, on page 599
Support for IPv4 and IPv6 unicast routing, non SVI IP unnumbered interface, and directed broadcast	Added support for IPv4 ans IPv6 unicast routing, non SVI IP unnumbered interface, and directed broadcast on N9336C-SE1 switches.	10.6(1)F	Guidelines and Limitations for IPv4, on page 31 Guidelines and limitations for IPv6, on page 79

Feature	Description	Changed in Release	Where Documented
Support for BGP, OSPFv2, OSPFv3, ISIS, PBR, and ECMP	Added support for BGP, OSPFv2, OSPFv3, ISIS, PBR and ECMP on N9336C-SE1 switches.	10.6(1)F	Guidelines and limitations for basic and advanced BGP, on page 299 Guidelines and limitations for OSPFv2, on page 118 Guidelines and limitations for OSPFv3, on page 169 Guidelines and limitations for IS-IS, on page 257 Guidelines and limitations for policy-based routing, on page 559
Support for URPF, HSRP, user-defined MAC address, vPC, and vPC peer gateway	Added support for URPF, HSRP, user-defined MAC address, vPC, and vPC peer gateway on N9336C-SE1 switches.	10.6(1)F	Guidelines and limitations for HSRP, on page 619

New and changed information



CHAPTER 2

Overview

This chapter contains these sections.

- Licensing Requirements, on page 7
- Supported Platforms, on page 7
- Layer 3 Unicast Routing, on page 7
- Routing Algorithms, on page 14
- Layer 3 Virtualization, on page 16
- Cisco NX-OS Forwarding Architecture, on page 16
- Guidelines and Limitations on the Cisco N9364E-SG2-Q Switch, on page 18
- Summary of Layer 3 Unicast Routing Features, on page 18
- Related Topics, on page 21

Licensing Requirements

See the Cisco NX-OS Licensing Guide and Cisco NX-OS Licensing Options Guide for Cisco NX-OS licensing recommendations and instructions to obtain and apply licenses.

Supported Platforms

See 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.

Layer 3 Unicast Routing

Layer 3 unicast routing involves two 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. Routing tables contain route information such as the IP destination address, the address of the next router, or the next hop. Routers use destination and next-hop associations to determine which router represents the next hop on the way to the 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. These metrics differ depending on the design of the routing algorithm used. See the Routing Metrics section.

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

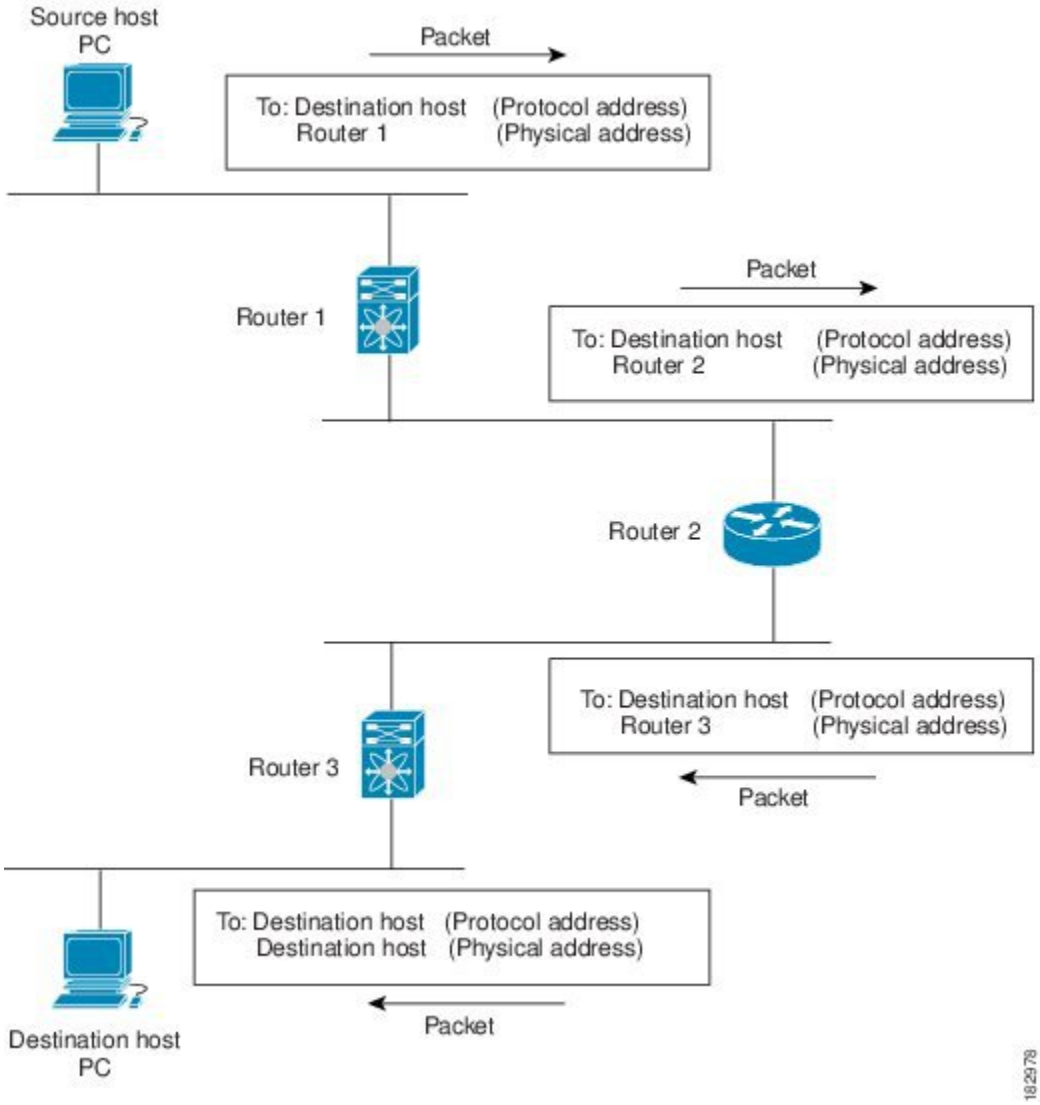
Packet Switching

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

The router checks destination IP in the routing table. If the router does not find the destination IP address in the routing table, it drops the packet. If the router finds the destination address in the routing table, the router 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 figure Figure 1: Packet Header Updates Through a Network , on page 9 for reference.

Figure 1: Packet Header Updates Through a Network



Route Metrics

Routing algorithms use a variety of metrics to determine the best route. Sophisticated routing algorithms can base route selection on multiple metrics.

Path Lengths

The path length is the most common routing metric. Some routing protocols allow you to assign arbitrary costs to each network link. The path length, in these protocols, is the sum of the costs for each traversed link.

Other routing protocols define the hop count. The hop count refers to 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 Delays

Routing delay is the length of time required to move a packet from a source to a destination through the internetwork. Routing delay depends on many factors such as

- bandwidth of intermediate network links
- port queues at each router along the way
- network congestion on all intermediate network links, and
- physical distance that the packet must 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 is preferable to a 1-Gigabit Ethernet link. Although the bandwidth is the maximum attainable throughput on a link, links with greater bandwidth do not always provide better routes than slower links. For example, a busy faster link might require more time to send a packet to its destination than a slower link.

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 usage 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. However, you can choose to 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, then Cisco NX-OS prefers the first loopback interface over any other interface type.

- If you have not configured a loopback interface, 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 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 (RIR) 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.

The Border Gateway Protocol (BGP) supports 4-byte AS numbers that can be represented in asplain and asdot notations:

- asplain—A decimal value notation where both 2-byte and 4-byte AS numbers are represented by their decimal value. For example, 65526 is a 2-byte AS number, and 234567 is a 4-byte AS number.
- asdot—An AS dot notation where 2-byte AS numbers are represented by their decimal value and 4-byte AS numbers are represented by a dot notation. For example, 2-byte AS number 65526 is represented as 65526, and 4-byte AS number 65546 is represented as 1.10.

The BGP 4-byte AS number capability is used to propagate 4-byte-based AS path information across BGP speakers that do not support 4-byte AS numbers.

Note RFC 5396 is partially supported. The asplain and asdot notations are supported, but the asdot+ notation is not.

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, see this 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, while 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. For the number of ECMP paths supported by each routing protocol, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.



Note ECMP does not guarantee equal load-balancing across all links. It guarantees only that a particular flow will choose one particular next hop at any point in time.

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 the Open Shortest Path First (OSPF) protocol 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, which prevents 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.



Note You are required to use route maps when you configure the redistribution of routing information.

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 picked over routes from another protocol with a higher administrative distance assigned.

Administrative Distances

An administrative distance is a rating of the trustworthiness of a routing information source. A higher value indicates a lower trustworthiness. 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. In stub routing, one or more end (stub) networks are connected to a remote router (the spoke) that is again connected to one or more distribution routers (the hub). The remote router is adjacent only to one or more distribution routers.

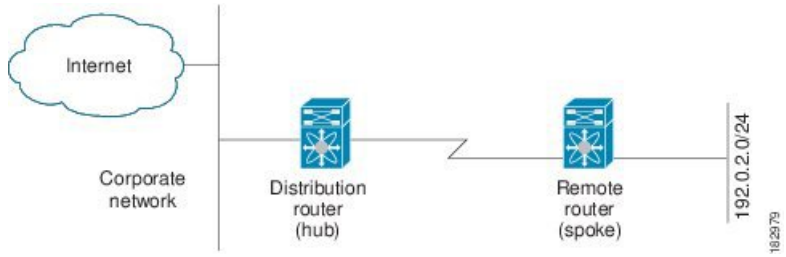
IP traffic can reach the remote router only through a distribution router. This type of configuration is commonly used in WAN topologies in which the distribution router is directly connected to a WAN. The distribution router can be connected to many more remote routers. Often, the distribution router 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 that informs 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. The figure Figure 2: Simple Hub-and-Spoke Network , on page 13 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 is always through the distribution router. A larger route table reduces 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 the Enhanced Interior Gateway Routing Protocol (EIGRP) supports stub routers.

Note Use the EIGRP stub routing feature only on stub devices. A stub device connects to the network core or distribution layer, and core transit traffic should not flow through it. IP traffic can reach the remote router only 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
- deals with topology changes
- determines the optimal route to a destination

Various types of routing algorithms exist. 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 large, constantly changing networks.

Dynamic routing protocols use dynamic routing algorithms that adjust to changing network circumstances by analyzing incoming routing update messages. Most routing protocols today use dynamic routing protocols. 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, which is a router to which all unrouteable packets are sent.

Interior and Exterior Gateway Protocols

Routing protocols that route traffic between autonomous systems are called exterior gateway protocols or interdomain protocols. Border Gateway Protocol (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.

Distance Vector Protocols

Distance vector protocols use distance vector algorithms that

- call for each router to send all or some portion of its routing table to its neighbors
- define routes by distance, such as number of hops to the destination, and
- direction, such as next-hop router.

These routes are then broadcast to the directly connected neighbor routers. Each router uses these updates to verify and update the routing tables. Distance vector algorithms are also known as Bellman-Ford algorithms.

To prevent routing loops, most distance vector algorithms use split horizon with poison reverse which means that the routes learned from an interface are set as unreachable and advertised back along the interface that they were learned on during the next periodic update. This process prevents the router from seeing its own route updates coming back.

Distance vector algorithms send updates at fixed intervals but can also send updates in response to changes in route metric values. These triggered updates can speed up the route convergence time. The Routing Information Protocol (RIP) is a distance vector protocol.

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) that 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 that it received to its neighbors because it wants to prevent an LSA update loop. Because the router floods the LSAs immediately after it receives them, the 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 the router's 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 less likely to cause routing loops than distance vector algorithms. However, link-state algorithms require more CPU power and memory than distance vector

algorithms and they 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 the Configuring Layer 3 Virtualization.

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

Cisco NX-OS Forwarding Architecture

The Cisco NX-OS forwarding architecture is responsible for processing all routing updates and populating the forwarding information to all modules in the chassis.

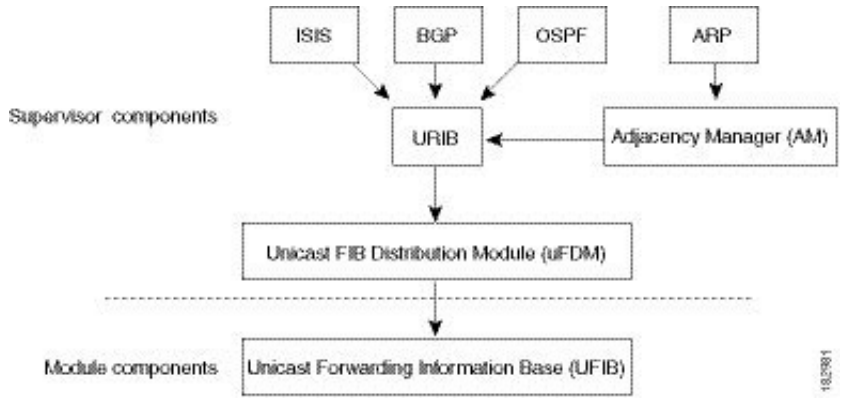
Unicast RIB

Unicast RIB exists on the active supervisor module of forwarding architecture that maintains

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

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

Figure 3: Cisco NX-OS Forwarding Architecture



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 exists on the active supervisor and maintains adjacency information for different protocols including ARP, Neighbor Discovery Protocol (NDP), and static configuration. 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 adjacency manager can trigger ARP requests to find a particular Layer 3 to Layer 2 mapping. The new mapping becomes available when the corresponding ARP reply is received and processed. For IPv6, the adjacency manager finds the Layer 3 to Layer 2 mapping information from NDP. For more information, see IPv6 Addresses, on page 57.

Unicast Forwarding Distribution Module

The unicast forwarding distribution module (FDM) exists on the active supervisor that

- distributes the forwarding path information from the unicast RIB and other sources
- programs the forwarding information that the unicast RIB generates into the hardware forwarding tables on the standby supervisor and the modules, and
- downloads the forwarding information base (FIB) information to newly inserted modules.

The unicast FDM 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 Forwarding Information Base (FIB) exists on supervisors and switching modules that

- builds the information used for the hardware forwarding engine

- receives route updates from the unicast FIB Distribution Module (FDM) and sends the information to the hardware forwarding engine, and
- controls the addition, deletion, and modification of routes, paths, and adjacencies.

Unicast FIBs are maintained on a per-VRF and per-address-family basis. Each VRF is configured with one unicast FIB for IPv4 and one for IPv6. 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.

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.

You also use the **show platform fib** or **show platform forwarding** commands to display details on hardware forwarding.

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 on the active supervisor. All packets that should be switched in the software or terminated go to the supervisor. The supervisor uses the information provided by the unicast RIB and the adjacency manager to make the forwarding decisions. The module is not involved in the software forwarding path.

Software forwarding is controlled by control plane policies and rate limiters. For more information, see the Cisco NX-OS 9000 Series NX-OS Security Configuration Guide.

Guidelines and Limitations on the Cisco N9364E-SG2-Q Switch

Starting with Cisco NX-OS Release 10.5(3)F, the Cisco Nexus 93C64E-SG2-Q switch supports these Layer 3 unicast and routing features.

- IPv4 & IPv6 routing
- Layer 3 ECMP
- ECMP Hashing

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.

IPv4 and IPv6 Addresses

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 IPv4 Addresses, on page 23 or IPv6 Addresses, on page 57.

IP Services

IP Services include Dynamic Host Configuration Protocol (DHCP) and Domain Name System (DNS Client) clients. For more information, see Configuring DNS.

OSPF

The Open Shortest Path First (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 that is connected to the link. The advertisements that contain this link information are called link-state advertisements. For more information, see OSPFv2, on page 111.

EIGRP

The Enhanced Interior Gateway Routing Protocol (EIGRP) 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. 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 EIGRP, on page 215.

IS-IS

The Intermediate System-to-Intermediate System (IS-IS) protocol is an intradomain Open System Interconnection (OSI) dynamic routing protocol specified in the International Organization for Standardization (ISO) 10589. The IS-IS routing protocol is a link-state protocol. IS-IS features are as follows:

- Hierarchical routing
- Classless behavior
- Rapid flooding of new information
- Fast Convergence
- Very scalable

For more information, see Configure IS-IS, on page 251.

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 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 preference to a route, origin of the route, community and others.

RIP

The Routing Information Protocol (RIP) is a distance-vector protocol that uses a hop count as its metric.

RIP is widely used for routing traffic in the global Internet and is an Interior Gateway Protocol (IGP), which means that it performs routing within a single autonomous system. For more information, see Configure RIP, on page 447.

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 virtual routing and forwarding (VRF).

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, on page 523.

Policy-Based Routing

Policy-based routing uses the Route Policy Manager to create policy route filters. These policy route filters can forward a packet to a specified next hop based on the source of the packet or other fields in the packet header. Policy routes can be linked to extended IP access lists so that routing might be based on protocol types and port numbers. For more information, see Configuring Policy-Based Routing.

First Hop Redundancy Protocols

First hop redundancy protocols (FHRP), such as the Hot Standby Router Protocol (HSRP) and the Virtual Router Redundancy Protocol (VRRP), allow 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 HSRP, see Configuring HSRP. For more information on VRRP, see Configure VRRP, on page 643.

Object Tracking

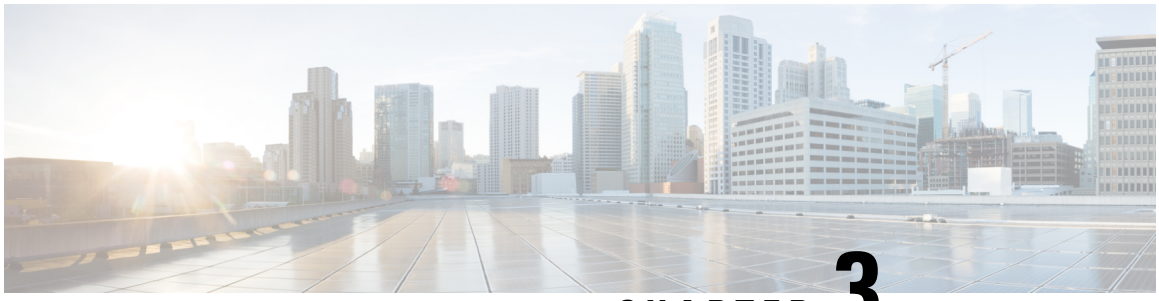
Object tracking allows you to track specific objects on the network, such as the interface line protocol state, IP routing, and route reachability, and take action when the tracked object's state changes.

This feature allows you to increase the availability of the network and shorten the recovery time if an object state goes down. For more information, see Configuring Object Tracking.

Related Topics

Feature Name	Feature Information
Layer 3 features	*Cisco NX-OS 9000 Series NX-OS Multicast Routing Configuration Guide* *Cisco Cisco NX-OS 9000 Series NX-OS High Availability and Redundancy Guide* Exploring Autonomous System Numbers: https://www.iana.org/numbers

Related Topics



CHAPTER 3

IPv4 Addresses

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 device.

This chapter includes the following sections:

- IPv4 Addresses, on page 23
- Virtualization support for IPv4, on page 31
- Prerequisites for IPv4, on page 31
- Guidelines and Limitations for IPv4, on page 31
- Default settings, on page 34
- Configure IPv4, on page 35
- Verify the IPv4 Configuration, on page 54
- Additional References, on page 55

IPv4 Addresses

An IP address is a unique identifier assigned to each network interface on a device, allowing communication with other hosts across a network. Configuring IP addresses on network interfaces enables those interfaces to send and receive packets within the network.

An interface can have one primary IP address, which is the main source and destination for packets generated by the device. You can also assign multiple secondary IP addresses to an interface to support multiple subnets or network requirements. All devices on the same interface must share the same primary IP address because outgoing packets always use this primary address as their source. Every IPv4 packet includes source and destination information taken from the configured IP addresses. For more information, see Multiple IPv4 Addresses.

A subnet mask is a 32-bit value that divides an IP address into the network and host portions. By applying a subnet mask, you specify the network to which an IP address belongs and isolate the host component. Subnet masks help IP networks distinguish between network IDs and host IDs, facilitating effective routing and addressing.

The IP feature on the device manages IPv4 packet handling and forwarding, including unicast and multicast route lookup, software-based access control list (ACL) forwarding, and duplicate address detection. It also oversees IP address configuration for network interfaces, static route management, and packet transmission and reception by IP clients.

Note Nexus switches drop packets that are sent to a null0 interface. If an IPv4 or IPv6 packet is sent to a null0 interface, Cisco Nexus 9000 switches do not respond with an ICMP or ICMPv6.

Multiple IPv4 Addresses

Cisco NX-OS allows you to configure multiple IP addresses for each interface. You can set any number of secondary addresses to meet different requirements. Common scenarios include:

- When you require more host IP addresses than a single subnet allows. For instance, if your subnetting scheme supports up to 254 hosts per logical subnet but you need 300 hosts on a single physical subnet, you can apply secondary IP addresses on the routers or access servers. This enables you to map two logical subnets onto one physical subnet.
- When two subnets of the same network are physically separated by another network, you can use a secondary address to unify them. In these cases, one network is effectively extended over another. Note that a subnet must not appear on more than one active interface on the router at any time.

Note If any device on a network segment uses a secondary IPv4 address, all other devices on that same interface must use a secondary address from the same network or subnet. Inconsistent application of secondary addresses within the segment can quickly result in routing loops.

LPM routing modes

By default, Cisco NX-OS programs routes in a hierarchical fashion to allow for the longest prefix match (LPM) on the device. However, you can configure the device for different routing modes to support more LPM route entries.

Theses tables list the LPM routing modes that are supported on Cisco Nexus 9000 Series switches.

LPM routing modes for Cisco Nexus 9200 platform switches

Table 2: LPM routing modes for Cisco Nexus 9200 platform switches

LPM routing mode	CLI command
Default system routing mode	
LPM dual-host routing mode	**system routing template-dual-stack-host-scale**
LPM heavy routing mode	**system routing template-lpm-heavy**

Note Cisco Nexus 9200 platform switches do not support the **system routing template-lpm-heavy** mode for IPv4 Multicast routes. Make sure to reset LPM's maximum limit to 0.

LPM routing modes for Cisco Nexus 9300 platform switches

Table 3: LPM routing modes for Cisco Nexus 9300 platform switches

LPM routing mode	Broadcom T2 mode	CLI command
Default system routing mode	3	
ALPM routing mode	4	**system routing max-mode l3**

LPM routing modes for Cisco Nexus 9300-EX/FX/FX2/FX3/GX platform switches

Table 4: LPM routing modes for Cisco Nexus 9300-EX/FX/FX2/FX3/GX platform switches

LPM routing mode	CLI command
LPM dual-host routing mode	**system routing template-dual-stack-host-scale**
LPM heavy routing mode	**system routing template-lpm-heavy**
LPM Internet-peering mode	**system routing template-internet-peering**

LPM routing modes for Cisco Nexus 9300-FX platform switches

LPM routing modes for Cisco Nexus 9300-fx2 platform switches

LPM routing modes for Cisco Nexus 9300-GX platform switches

LPM routing modes for Cisco Nexus 9500 platform switches with 9700-ex and 9700-fx line cards

Table 5: LPM routing modes for Cisco Nexus 9500 platform switches with 9700-ex and 9700-fx line cards

LPM routing mode	Broadcom T2 mode	CLI command
Default system routing mode	3 (for line cards); 4 (for fabric modules)	
Max-host routing mode	2 (for line cards); 3 (for fabric modules)	**system routing max-mode host**
Nonhierarchical routing mode	3 (for line cards); 4 with max-l3-mode option (for line cards)	**system routing non-hierarchical-routing [max-l3-mode]**
64-bit ALPM routing mode	Submode of mode 4 (for fabric modules)	**system routing mode hierarchical 64b-alpm**

LPM routing mode	Broadcom T2 mode	CLI command
LPM heavy routing mode		**system routing template-lpm-heavy** **Note** This mode is supported only for Cisco Nexus 9508 switches with the 9732C-EX line card.
LPM Internet-peering mode		**system routing template-internet-peering** **Note** This mode is supported only for the following Cisco Nexus 9500 Platform Switches: • Cisco Nexus 9500 platform switches with 9700-EX line cards. • Cisco Nexus 9500-FX platform switches (Cisco NX-OS release 7.0(3)I7(4) and later) • Cisco 9500-R platform switches (Cisco NX-OS release 9.3(1) and later)
LPM dual-host routing mode		

LPM routing modes for Cisco Nexus 9500-R platform switches with 9600-R line cards

Table 6: LPM Routing Modes for Cisco Nexus 9500-R Platform Switches with 9600-R Line Cards

LPM routing Mode	CLI command
LPM Internet-peering mode	**system routing template-internet-peering** (Cisco NX-OS release 9.3(1) and later)

Host to LPM Spillover

Beginning with Cisco NX-OS Release 7.0(3)I5(1), you can store host routes on the longest prefix match (LPM) table to achieve a larger host scale.

In a LPM mode, the switch supports fewer host routes. If you add more host routes than the supported scale, the excess routes from the host table occupy space in the LPM table, which reduces the total number of LPM routes available. This feature is supported on Cisco Nexus 9300 and 9500 platform switches.

In the default system routing mode, Cisco Nexus 9300 platform switches are set for higher host scale and fewer LPM routes, allowing the LPM space to be used for additional host routes. On Cisco Nexus 9500 platform switches, this feature is supported on line cards only in the default system routing and nonhierarchical routing modes. Fabric modules do not support this feature.

Address Resolution Protocol

Networking devices and Layer 3 switches use Address Resolution Protocol (ARP) to map IP addresses (network layer) to Media Access Control (MAC) address (physical layer) to enable IP packets to be sent across networks.

Before a device sends a packet to another device, it looks in its own ARP cache to see if there is a MAC address and corresponding IP address for the destination device. If there is no entry, the source device sends a broadcast message to every device on the network.

Each device compares the IP address to its own. Only the device with the matching IP address replies to the device that sends the data with a packet that contains the MAC address for the device. The source device adds the destination device 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 device lies on a remote network that is beyond another device, the process is the same except that the device 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 packet, the default gateway broadcasts the destination IP address over the networks connected to it. The device on the destination device network uses ARP to obtain the MAC address of the destination device and delivers the packet. ARP is enabled by default.

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

ARP cache

The Address Resolution Protocol (ARP) cache is a table maintained by network devices such as routers and switches to store mappings between IP addresses and their corresponding MAC hardware addresses.

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

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

- Every device on a network updates its tables as addresses are broadcast.

Static and dynamic entries in the ARP Cache

Static route

Static routing requires you to configure the IP addresses, subnet masks, gateways, and corresponding MAC addresses for each interface of each device.

Static routing requires more work to maintain the route table. You must update the table each time you add or change routes.

Dynamic route

Dynamic routing uses protocols that enable the devices in a network to exchange routing table information with each other.

Dynamic routing is more efficient than static routing because the route table is automatically updated unless you add a time limit to the cache. The default time limit is 25 minutes, but you can modify the time limit if the network has many routes that are added and deleted from the cache.

Devices that do not use ARP

When a network is segmented into two parts, a bridge connects these segments and filters traffic based solely on MAC addresses. The bridge maintains its own MAC address table to manage this filtering. In contrast, a device maintains an ARP cache that maps both IP addresses and their corresponding MAC addresses.

Passive hubs serve as central connection points that physically link devices within a network. Operating at Layer 1 (the physical layer), they broadcast incoming messages to all ports without maintaining any address table.

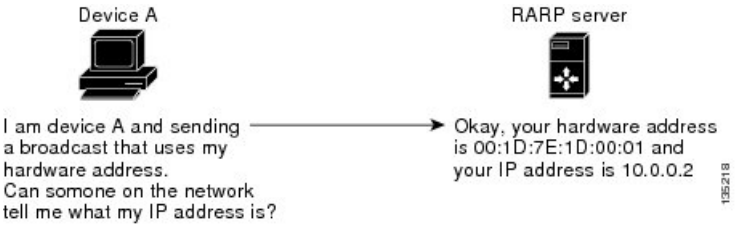
Layer 2 switches determine which port of a device receives a message that is sent only to that port. However, Layer 3 switches are devices that build an ARP cache (table).

Reverse ARP

Reverse ARP (RARP) works the same way as ARP, except that the RARP request packet requests an IP address instead of a MAC address. RARP is defined by RFC903.

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 Figure 5: Reverse ARP, on page 29 figure shows how RARP works.

Figure 5: Reverse ARP



RARP has several limitations. Because of these limitations, most businesses use Dynamic Host Configuration Protocol (DHCP) to assign IP addresses dynamically. DHCP is cost effective and requires less maintenance than RARP.

These 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 is a feature that allows a device, typically a router, to answer ARP requests on behalf of another device. This allows a host to communicate with devices on remote subnets without needing to configure routing or a default gateway.

These are the advantages of proxy ARP:

- Proxy ARP enables a device that is physically located on one network to be logically part of a different physical network connected to the same device or firewall.
- Proxy ARP allows you to hide a device with a public IP address on a private network behind a router, yet still have the device 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 devices on a subnet reach remote subnets without configuring routing or a default gateway.

When devices 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 devices 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 device and it receives an ARP request, it identifies the request as a request for a system that is not on the local LAN. The device responds as if it is the remote destination for which the broadcast is addressed, with an ARP response that associates the device’s MAC address with the remote destination's IP address. The local device 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 device. By default, proxy ARP is disabled.

Local proxy ARP

Local proxy ARP enables a device 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 device to which they are connected.

Gratuitous ARP

Gratuitous ARP sends a request with an identical source IP address and a destination IP address to detect duplicate IP addresses.

Cisco NX-OS supports enabling or disabling gratuitous ARP requests or ARP cache updates.

Periodic ARP refresh on MAC delete

The ARP process tracks the MAC deletes and sends the periodic ARP Refresh on the L3 VLAN interface in a configured interval of time for the configured count. If the MAC is learned, ARP process stops sending the periodic ARP Refreshes.

For more information, see Configuring periodic ARP refresh on MAC delete for SVIs, on page 47.

ARP refresh on MAC Delete

MAC addresses are deleted or flushed when triggers such as interface flaps, clearing the MAC address table, or STP TCN notifications occur. Once the MAC addresses are deleted, the ARP process initiates ARP Refresh for all adjacencies associated with these MAC addresses. These ARP Refreshes are subject to CoPP limits, which can prevent the ARP process from effectively relearning all ARP entries.

Beginning with Cisco NX-OS Release 10.5(3)F, instead of sending Refresh retries at 2, 4, 8 seconds and deleting after 16 seconds (if no response is received), the ARP process sends the Refreshes at 2, 4, 8, and 8 seconds, and deletes after an additional 8 seconds (one extra Refresh). This change improves ARP learning with scaled ARP entries, even if some ARP response packets are subject to CoPP limits.

Glean throttles

Glean throttling is a mechanism where, if an Address Resolution Protocol (ARP) request for the next-hop IP address is not resolved by a line card when forwarding incoming IP packets, the line card forwards those packets to the supervisor for resolution. The supervisor resolves the MAC address for the next hop and programs 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.

Path MTU discovery

Path maximum transmission unit (MTU) discovery is a method for maximizing the use of available bandwidth in the network between the endpoints of a TCP connection.

It is described in RFC 1191. Existing connections are not affected when this feature is turned on or off.

ICMP

You can use the Internet Control Message Protocol (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 it 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.
- The ICMP redirect feature is not supported on the Cisco N93-C64E-SG2-Q switch platform. If ICMP redirect is currently enabled on the Cisco N93-C64E-SG2-Q switch, it must be disabled.

Virtualization support for IPv4

IPv4 supports virtual routing and forwarding (VRF) instances.

Prerequisites for IPv4

IPv4 has the following prerequisites:

- IPv4 can only be configured on Layer 3 interfaces.

Guidelines and Limitations for IPv4

IPv4 has the following configuration guidelines and limitations:

- Cisco Nexus 9300-EX and Cisco Nexus 9300-FX2 platform switches configured for internet-peering mode might not have sufficient hardware capacity to install full IPv4 and IPv6 Internet routes simultaneously.
- You can configure a secondary IP address only after you configure the primary IP address.
- Local proxy ARP is not supported for an interface with more than one HSRP group that belongs to multiple subnets.
- The **ip proxy-arp** command is not supported in a VXLAN EVPN Fabric specifically on an SVI which is enabled with **fabric forwarding mode anycast-gateway** .
- Local proxy ARP does not work on the VLAN when you disable ARP suppression on the VLAN.
- For Cisco Nexus 9500 platform switches with -R line cards, internet-peering mode is only intended to be used with the prefix pattern as distributed in the global internet routing table. In this mode, other prefix distributions/patterns can operate, but not predictably. As a result, maximum achievable LPM/LEM scale is reliable only when the prefix patterns are actual internet prefix patterns. In Internet-peering mode, if route prefix patterns other than those in the global internet routing table are used, the switch might not successfully achieve documented scalability numbers.
- When **system routing template-internet-peering** is configured, multicast traffic is not supported. However, it is still possible to have feature PIM and PIM configurations in the switch.
- LPM heavy routing mode is supported on Cisco Nexus **9500** series switches with **9700**-EX, -FX, and -GX series modules.
- Beginning with Cisco NX-OS Release 10.2(3)F, syslog will be printed when IPv4 redirect message is triggered based on the configured interval.
- Beginning with Cisco NX-OS Release 10.3(1)F, static routing is supported on the Cisco Nexus 9808 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, static routing is supported on the following switches and line cards:
  - Cisco Nexus 9804 switches.
  - Cisco Nexus X98900CD-A, and X9836DM-A line cards with 9804, and 9808 platform switches.
- Beginning with Cisco NX-OS Release 10.3(1)F, dynamic routing is supported on the Cisco Nexus 9808 platform switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, dynamic routing is supported on Cisco Nexus X98900CD-A and X9836DM-A line cards with 9808 and 9804 switches.
- Beginning with Cisco NX-OS Release 10.4(2)F, static routing is supported on Cisco Nexus 9232E-B1 switch.
- Beginning with Cisco NX-OS Release 10.2(4)M, periodic ARP Refresh on MAC delete support is provided on Cisco Nexus 9000 Series platform switches with the following limitations:
  - During configuration of the **ip arp refresh-adj-on-mac-delete retry** command, ARP process does not trigger Refresh although ARP is learned and MAC is not learned. It tries to send periodic ARP Refreshes on MAC delete/flush.
  - The periodic ARP Refresh behavior is triggered for the MAC's deletion after configuring the **ip arp refresh-adj-on-mac-delete retry** command.

- The trigger for this periodic ARP Refresh is MAC delete. This feature does not address the MAC learn miss on receiving burst packets.
- During configuring, you must choose the right count and interval based on the scale/network requirements.

- Beginning with Cisco NX-OS Release 10.4(1)F, out of subnet ARP resolution support is provided on Cisco Nexus 9000 Series platform switches for the following L3 interfaces:
  - Ethernet
  - Sub-interfaces
  - Port-channel
  - FEX
  - IP unnumbered interface



Note
- The out of subnet ARP resolution feature is not supported on SVI L3 interfaces and on VPC or HSRP or VXLAN deployments.

- Beginning with Cisco NX-OS Release 10.4(2)F, the **ip arp cache intf-limit** configuration is supported to limit the ARP cache entries per interface on Cisco NX-OS devices with the following capabilities:
  - Supported on global and interface modes. However, interface mode configuration takes the precedence over global mode.
  - Supported only on the following L3 interfaces:
    - SVI
    - SVI Unnumbered Interfaces
  - Not supported on the following L3 interfaces:
    - Ethernet
    - Subinterfaces
    - Port-channel
    - Unnumbered interfaces
  - If the configuration is applied to non-supporting interfaces, this configuration will be applied to the global mode.

- Beginning with Cisco NX-OS Release 10.4(2)F, the following IPv4 Layer-3 features are supported on the Cisco Nexus 9232E-B1 switches:
  - ARP
  - ECMP
  - IPv4 unicast routes

    - IPv4 host routes

- Beginning with Cisco NX-OS Release 10.4(2)F, you can configure all NX-OS features using Class E IP address. If the IPv4 address space is exhausted, you can configure the features using the Class E IP address. The following switches and line cards are supported for Class E IP address:
    - Cisco Nexus 92348GC-X
    - Cisco Nexus 9300-EX/FX/FX2/FX3/H2R/H1/GX/GX2
    - Cisco Nexus 9300C
    - Cisco Nexus 9700-EX/FX/GX line cards
- Beginning with Cisco NX-OS Release 10.6(1)F, the following IPv4 Layer-3 features are supported on the Cisco Nexus N9336C-SE1 switches:
    - IPv4 unicast routes
    - IPv4 host routes
    - IPv4 Neighbor Discovery
    - ARP
    - IP directed broadcast
    - Non SVI IP unnumbered interface
- Beginning with Cisco NX-OS Release 10.6(2)F, you can configure hardware forwarding of IP directed broadcasts using new **hw-assist** keyword in IP directed broadcast configuration.
    - This feature is supported only on L3 topology
    - This feature is not supported on VXLAN
    - Keyword hw-assist is supported only on SVIs
    - This feature is supported on Cisco Nexus 9300-FX/FX2/FX3/GX/GX2/HX based TORs. It is not supported for EOR.
- Beginning with Cisco NX-OS Release 10.6(2)F, following features are supported on Cisco Nexus N9324C-SE1U and N9348Y2C6D-SE1U platform switches
    - Non SVI IP unnumbered interface
    - IP directed broadcast
- Beginning with Cisco NX-OS Release 10.6(2n), Layer 3 support for IPv4 is supported on the Cisco N9164E-NS4-O switch.

Default settings

The table below lists the default settings for IP parameters.

Parameters	Default
ARP timeout	1500 seconds
Proxy ARP	Disabled

Configure IPv4



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.

Configure IPv4 address

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

Procedure

Step 1 Use the **configure terminal** command to global configuration mode.

Example:

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

Step 2 Use the **interface ethernet** *number* command to enter interface configuration mode.

Example:

```
switch(config)# interface ethernet 2/3
switch(config-if)#
```

Step 3 Use the **ip address** *ip-address/length* [*secondary*] command to specify a primary or secondary IPv4 address for an interface.

Example:

```
switch(config-if)# ip address
192.2.1.1 255.0.0.0
```

- 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, which is the 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 must be no space between the IP address and the slash.

Step 4 (Optional) Use the **show ip interface** command to display interfaces configured for IPv4.

Example:

```
switch(config-if)# show ip interface
```

Step 5 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config-if)# copy running-config
startup-config
```

Configure multiple IP address

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

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface ethernet** *number* command to enter interface configuration mode.

Example:

```
switch(config)# interface ethernet 2/3
switch(config-if)#
```

Step 3 Use the **ip address** *ip-address/length* [*secondary*] command to specify a the configured address as a secondary IPv4 address.

Example:

```
switch(config-if)# ip address
192.168.1.1 255.0.0.0 secondary
```

Step 4 (Optional) Use the **show ip interface** command to see interfaces configured for IPv4.

Example:

```
switch(config-if)# show ip interface
```

Step 5 (Optional) Use the **copy running-config startup-config** command to save this configuration.

Example:

```
switch(config-if)# copy running-config
startup-config
```

Configure max-host routing mode

By default, Cisco NX-OS programs routes in a hierarchical fashion (with fabric modules that are configured to be in mode 4 and line card modules that are configured to be in mode 3), which allows for longest prefix match (LPM) and host scale on the device.

You can modify the default LPM and host scale to program more hosts in the system, as might be required when the node is positioned as a Layer-2 to Layer-3 boundary node.

Note If you want to further scale the entries in the LPM table, see the Configuring Nonhierarchical Routing Mode (Cisco Nexus 9500 Series Switches Only) section to configure the device to program all the Layer 3 IPv4 and IPv6 routes on the line cards and none of the routes on the fabric modules.

Note This configuration impacts both the IPv4 and IPv6 address families.

Note For the max-host routing mode scale numbers, refer to the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **system routing max-mode host** command to put the line cards in Broadcom T2 mode 2 and the fabric modules in Broadcom T2 mode 3 to increase the number of supported hosts.

Example:

```
switch(config)# system routing max-mode host
```

Step 3 (Optional) Use the **show forwarding route summary** command to see the LPM routing mode.

Example:

```
switch(config)# show forwarding route summary
```

Step 4 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Step 5 Use the **reload** command to root the entire device.

Example:

```
switch(config)# reload
```

Configure nonhierarchical routing mode (Cisco Nexus 9500 platform switches only)

If the host scale is small (as in a pure Layer 3 deployment), we recommend programming the longest prefix match (LPM) routes in the line cards to improve convergence performance. Doing so programs routes and hosts in the line cards and does not program any routes in the fabric modules.

Note This configuration impacts both the IPv4 and IPv6 address families.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **system routing non-hierarchical-routing** [**max-l3-mode**] command to put the line cards in Broadcom T2 mode 3.

Example:

```
switch(config)# system routing
non-hierarchical-routing max-l3-mode
```

Alternatively, Broadcom T2 mode 4 if you use the **max-l3-mode** option to support a larger LPM scale. As a result, all of the IPv4 and IPv6 routes will be programmed on the line cards rather than on the fabric modules.

Step 3 (Optional) Use the **show forwarding route summary** command to see the LPM mode.

Example:

```
switch(config)# show forwarding route
      summary
      Mode 3:
      120K IPv4 Host table
      16k LPM table (> 65 < 127 1k entry
      reserved)
      Mode 4:
      16k V4 host/4k V6 host
      128k v4 LPM/20K V6 LPM
```

Step 4 Use the **copy running-config startup-config** command to saves this configuration change.

Example:

```
switch(config)# copy running-config
startup-config
```

Step 5 use the **reload** command to reboot the entire device.

Example:

```
switch(config)# reload
```

Configure 64-Bit ALPM routing mode (Cisco Nexus 9500 platform switches only)

You can use the 64-bit algorithmic longest prefix match (ALPM) feature to manage IPv4 and IPv6 route table entries. In 64-bit ALPM routing mode, the device can store more route entries. In this mode, you can program one of the following:

- 80,000 IPv6 entries and no IPv4 entries
- No IPv6 entries and 128,000 IPv4 entries
- *x* IPv6 entries and y IPv4 entries, where $2x + y <= 128,000$

Note This configuration impacts both the IPv4 and IPv6 address families.

Note For the 64-bit ALPM routing mode scale numbers, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 use the [**no**] **system routing mode hierarchical 64b-alpm** command to program all IPv4 and IPv6 LPM routes with a mask length less than or equal to 64 in the fabric module.

Example:

```
switch(config)# system routing mode
hierarchical 64b-alpm
```

All host routes for IPv4 and IPv6 and all LPM routes with a mask length of 65–127 are programmed in the line card.

Step 3 (Optional) Use the **show forwarding route summary** command to see the LPM mode.

Example:

```
switch(config)# show forwarding route
summary
```

Step 4 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config
startup-config
```

Saves this configuration change.

Step 5 Use the **reload** command to reboot the entire device.

Example:

```
switch(config)# reload
```

Configure ALPM routing mode (Cisco Nexus 9300 platform switches only)

You can configure Cisco Nexus 9300 platform switches to support more LPM route entries.

This configuration impacts both the IPv4 and IPv6 address families.

- For ALPM routing mode scale numbers, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **system routing max-mode l3** command to put the device in Broadcom T2 mode 4 to support a larger LPM scale.

Example:

```
switch(config)# system routing max-mode l3
```

Step 3 (Optional) Use the **show forwarding route summary** command to see the LPM mode.

Example:

```
switch(config)# show forwarding
route summary
```

Step 4 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config
startup-config
```

Step 5 Use the **reload** command to reboot the entire device.

Example:

```
switch(config)# reload
```

Configure LPM heavy routing mode (Cisco Nexus 9200 and 9300-EX platform switches and 9732C-EX line card only)

Beginning with Cisco NX-OS Release 7.0(3)I4(4), you can configure LPM heavy routing mode in order to support more LPM route entries. Only the Cisco Nexus 9200 and 9300-EX platform switches and the Cisco Nexus 9508 switch with an 9732C-EX line card support this routing mode.

Note This configuration impacts both the IPv4 and IPv6 address families.

Note For LPM heavy routing mode scale numbers, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **system routing template-lpm-heavy** command to put the device in LPM heavy routing mode to support a larger LPM scale.

Example:

```
switch(config)# system routing template-lpm-heavy
```

Step 3 (Optional) Use the **show system routing mode** command to see the LPM routing mode.

Example:

```
switch(config)# show system routing mode
                      Configured System Routing Mode: LPM Heavy
                      Applied System Routing Mode: LPM Heavy
```

Displays the LPM routing mode.

Step 4 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Step 5 Use the **reload** command to reboot the entire device.

Example:

```
switch(config)# reload
```

Configuring LPM Internet-Peering Routing Mode

Beginning with Cisco NX-OS Release 7.0(3)I6(1), you can configure LPM Internet-peering routing mode in order to support IPv4 and IPv6 LPM Internet route entries. This mode supports dynamic Trie (tree bit lookup) for IPv4 prefixes (with a prefix length up to /32) and IPv6 prefixes (with a prefix length up to /83).

Beginning with Cisco NX-OS Release 9.3(1), Cisco Nexus 9500-R platform switches support this routing mode.

Note This configuration impacts both the IPv4 and IPv6 address families.

Note For LPM Internet-peering routing mode scale numbers, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide. Cisco Nexus 9500-R platform switches in LPM Internet-peering mode scale out predictably only if they use internet-peering prefixes. If Cisco Nexus 9500-R platform switches use other prefix patterns, it might not achieve documented scalability numbers.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **system routing template-internet-peering** command to put the device in LPM Internet-peering routing mode to support IPv4 and IPv6 LPM Internet route entries.

Example:

```
switch(config)# system routing template-internet-peering
```

Step 3 (Optional) Use the **show system routing mode** command to see the LPM mode.

Example:

```
switch(config)# show system routing mode
                        Configured System Routing Mode: Internet Peering
                        Applied System Routing Mode: Internet Peering
```

Step 4 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Step 5 Use the **reload** command to reboot the entire device.

Example:

```
switch(config)# reload
```

Configure LPM dual-host routing mode

Beginning with Cisco NX-OS Release 7.0(3)I5(1), you can configure LPM dual-host routing mode in order to increase the ARP/ND scale to double the default mode value. Only the Cisco Nexus 9200 and 9300-EX platform switches support this routing mode.

Beginning with Cisco NX-OS Release 10.3(1)F, the **system routing template-dual-stack-host-scale** profile supports Multicast and VXLAN on Cisco Nexus 9300-FX3/GX/GX2B ToR switches, and Nexus 9408 switches.

Note Ensure that the **system routing template-dual-stack-host-scale** profile is not used with BGW.

Note This configuration impacts both the IPv4 and IPv6 address families.

Note For LPM dual-host routing mode scale numbers, see the **Cisco Nexus 9000 Series NX-OS Verified Scalability Guide**.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **system routing template-dual-stack-host-scale** command to put the device in LPM dual-host routing mode to support a larger ARP/ND scale.

Example:

```
switch(config)# system routing template-dual-stack-host-scale
                       Warning: The command will take effect after next reload.
                       Note: This requires copy running-config to startup-config before switch
reload.
```

Step 3 (Optional) Use the **show system routing mode** command to see the LPM routing mode.

Example:

```
switch(config)# show system routing mode
```

Step 4 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Step 5 Use the **reload** command to reboot the entire device.

Example:

```
switch(config)# reload
```

Configure a Static ARP Entry

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

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface ethernet** *number* command to enter interface configuration mode.

Example:

```
switch(config)# interface ethernet 2/3
switch(config-if)#
```

Step 3 Use the **ip arp address** *ip-address mac-address* command to associate an IP address with a MAC address as a static entry.

Example:

```
switch(config-if)# ip arp 192.168.1.1
0019.076c.1a78
```

Step 4 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config
startup-config
```

Configure proxy ARP

Configure proxy ARP on the device to determine the media addresses of hosts on other networks or subnets.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface ethernet** *number* command to enter interface configuration mode.

Example:

```
switch(config)# interface ethernet 2/3
switch(config-if)#
```

Step 3 Use the **ip proxy arp** command to enable proxy ARP on the interface.

Example:

```
switch(config-if)# ip proxy arp
```

Step 4 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config
startup-config
```

Configure local proxy ARP on ethernet interfaces

You can configure local proxy ARP on Ethernet interfaces.

Before you begin

Local proxy ARP is supported only on Layer 3 Ethernet interfaces. It is not supported on Layer 2 Ethernet interfaces.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface ethernet** *number* command to enter interface configuration mode.

Example:

```
switch(config)# interface ethernet 2/3
switch(config-if)#
```

Step 3 Use the [**no**]**ip local-proxy-arp** command to enable Local Proxy ARP on the interface.

Example:

```
switch(config-if)# ip local-proxy-arp
```

Step 4 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config startup-config
```

Configuring local proxy ARP on SVIs

You can configure local proxy ARP on SVIs, and beginning with Cisco NX-OS Release 7.0(3)I7(1), you can suppress ARP broadcasts on corresponding VLANs.

Before you begin

If you are planning to suppress ARP broadcasts, configure the double-wide ACL TCAM region size for ARP/Layer 2 Ethertype using the hardware access-list tcam region arp-ether 256 double-wide command, save the configuration, and reload the switch. (For more information, see the Configuring ACL TCAM Region Sizes section in the Cisco Nexus 9000 Series NX-OS Security Configuration Guide.)

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface vlan vlan-id** command to create a VLAN interface and enters the configuration mode for the SVI.

Example:

```
switch(config)# interface vlan 5
switch(config-if)#
```

Step 3 Use the [**no**] **ip local-proxy-arp** [**reachable**] command to enable local proxy ARP on SVIs.

Example:

```
switch(config-if)# ip local-proxy-arp reachable
```

Use the **reachable** option to enables enhanced proxy ARP which ensures ARP responses are sent only if the destination host is known or reachable. If unknown, a glean ARP request is generated to discover the destination.

Note Beginning with Cisco NX-OS Release 10.6(3)F, the **no-hw-flooding** option is deprecated and its functionality is now included as part of the default **ip local-proxy-arp** command.

Step 4 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config-if)# copy running-config startup-config
```

Configuring periodic ARP refresh on MAC delete for SVIs

Beginning with Cisco NX-OS Release 10.2(4)M, you can configure periodic ARP Refresh on MAC delete for SVIs.

By default this command is disabled. This command must be configured under the SVI for the periodic ARP Refreshes to learn the MACs from the ARP response packet for the silent hosts on MAC delete/flush.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface vlan vlan-id** command to create a VLAN interface and enter the configuration mode for the SVI.

Example:

```
switch(config)# interface vlan 5
                        switch(config-if)#
```

Step 3 Use the [**no**] **ip arp refresh-adj-on-mac-delete retry** [**count** <*frequency count*>] [**interval** <*interval in sec*>] command to configure the ARP Refreshes to learn the MACs from the ARP response packet for the silent hosts on MAC delete/flush.

Example:

```
switch(config-if)# ip arp refresh-adj-on-mac-delete retry count 3 interval 15
                        switch(config-if)#
```

- <*frequency count*>: The range is 1–3. The default is 3.
- <*interval in sec*>: The range is 1–60 seconds. The default is 15 seconds.

Note If the interval is greater than 3/4th of ARP Refresh time, this command is rejected with the below message:

```
ARP refresh will be sent earlier to the interval due to ARP timeout
configuration. This configuration is not useful.
```

Step 4 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config-if)# copy running-config startup-config
                        switch(config-if)#
```

Configure gratuitous ARP

You can configure gratuitous ARP on an interface.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface ethernet** *number* command to enter interface configuration mode.

Example:

```
switch(config)# interface ethernet 2/3
switch(config-if)#
```

Step 3 Use the **ip arp gratuitous** {**request** | **update**] command to enable gratuitous ARP on the interface.

Example:

```
switch(config-if)# ip arp gratuitous
request
```

Gratuitous ARP is enabled by default.

Step 4 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config
startup-config
```

Configure out of subnet ARP resolution

Beginning with Cisco NX-OS Release 10.4(1)F, you can enable or disable out of subnet ARP resolution using the **ip arp outside-subnet** command.

This command is available on both global and interface mode. There is no impact on config-replace and dual stage commit when this command is enabled.

Note When this command is enabled, downgrade from Cisco NX-OS Release 10.4(1)F is restricted, and user will be prompted with an error message to remove the out of subnet ARP resolution configuration before proceeding for downgrade.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **[no] ip arp outside-subnet** command to enable or disables the ARP out of subnet packet transaction on connected host.

Example:

```
switch(config)# ip arp outside-subnet
```

Step 3 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Configure ARP cache limit per SVI interface

Beginning from Cisco NX-OS Release 10.4(2)F, you can set the number of maximum ARP cache entries to be allowed per SVI interface on the Cisco NX-OS devices. This configuration is supported on both global and interface modes.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface vlan** *vlan-id* command to create a VLAN interface and enters the configuration mode for the SVI.

Example:

```
switch(config)# interface vlan 5
 switch(config-if)#
```

Step 3 Use the **[no] ip arp cache intf-limit** *value* command to configure the set limit of ARP cache entries for the SVI interface

Example:

```
switch(config-if)# ip arp cache intf-limit 50000
 switch(config-if)#
```

Range of valid ARP entries is 1-128000.

intf-limit: Specifies the number of valid dynamic ARP entries per interface.

The **no** form of this command removes the configuration.

Step 4 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Configure path MTU discovery

You can configure path MTU discovery.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **ip tcp path-mtu-discovery** command to enable path MTU discovery.

Example:

```
switch(config)# ip tcp
path-mtu-discovery
```

Step 3 (Optional) Use the **copy running-config startup-config** commad to save this configuration change.

Example:

```
switch(config)# copy running-config
startup-config
```

Configure IP directed broadcasts

IP directed broadcast feature forwards a packet from a node, which is not part of the destination subnet, at the link-layer in the destination subnet post IP routing

A device that is not directly connected to its destination subnet forwards an IP directed broadcast in the same way it forwards unicast IP packets destined to a host on that subnet. When a directed broadcast packet reaches a device that is directly connected to its destination subnet, that packet is broadcast on the destination subnet.

The destination MAC address is rewritten as broadcast address and the packet is forwarded 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 are broadcasted on that subnet. You can optionally filter those broadcasts through an IP access list such that only those packets that pass through the access list are broadcasted on the subnet.

Beginning with Cisco NX-OS Release 10.6(2)F, new keyword hw-assist is introduced that enables hardware forwarding of IP directed broadcasts.

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

Procedure

Use the **ip directed-broadcast**[*acl* | **hw-assist**] command to enable the translation of a directed broadcast to physical broadcasts.

Example:

```
switch(config-if) # ip directed-broadcast
```

You can optionally filter those broadcasts through an IP access list.

CLI **ip directed-broadcast** enables software forwarding of IP directed broadcasts.

- acl: filters the IP directed broadcast packets through the specified IP access list.
- hw-assist: enables hardware forwarding of IP directed broadcasts.

You can either use the **ip directed-broadcast** *acl* command or the **ip directed-broadcast hw-assist** command on an interface. If you configure both the commands, the latest configuration takes precedence.

Configuring IP glean throttling

We recommend that you configure IP glean throttling 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.

SUMMARY STEPS

1. Use the **configure terminal** command to enter global configuration mode.
2. Use the [**no**] **hardware ip glean throttle** command to enable IP glean throttling.
3. (Optional) Use the **copy running-config startup-config** command to save this configuration change.

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	Use the **configure terminal** command to enter global configuration mode.
	Example:
	`switch# configure terminal`
	`switch(config)#`	

	Command or Action	Purpose
Step 2	Use the [**no**] **hardware ip glean throttle** command to enable IP glean throttling. **Example:** `switch(config) # hardware ip glean throttle`	
Step 3	(Optional) Use the **copy running-config startup-config** command to save this configuration change. **Example:** `switch(config)# copy running-config startup-config`	

Configure the hardware IP glean throttle maximum

You can limit the maximum number of drop adjacencies that are installed in the Forwarding Information Base (FIB).

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **hardware ip glean throttle maximum** *count* command to configure the number of drop adjacencies that are installed in the FIB.

Example:

```
switch(config) # hardware ip glean
throttle maximum 2134
```

Step 3 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config
startup-config
```

Configure the hardware IP glean throttle timeout

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

SUMMARY STEPS

1. Use the **configure terminal** command to enter global configuration mode.

2. Use the [**no**] **hardware ip glean throttle maximum timeout** *timeout-in-seconds* command to configure the timeout for the installed drop adjacencies to remain in the FIB.
3. (Optional) Use the **copy running-config startup-config** command save this configuration change.

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	Use the **configure terminal** command to enter global configuration mode.
	Example:
	`switch# configure terminal`
	`switch(config)#`	

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

Configure 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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **ip source** {**ethernet** *slot/port* | **loopback** *number* | **port-channel** *number*} **icmp-errors** command tp configure an interface IP address for the ICMP source IP field to route ICMP error messages.

Example:

```
switch(config)# ip source loopback 0
icmp-errors
```

Step 3 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config
startup-config
```

Configure IPv4 redirect syslog

To enable/disable the IPv4 redirect syslog or change the logging interval, use the below CLIs:

Note By default, redirecting syslog will be enabled.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **ip redirect syslog** [<*value*>] command to configure the syslog for excessive IP redirect messages.

Example:

```
switch(config)# ip redirect syslog 60
                        switch(config)#
```

- **ip redirect syslog**: Enables the syslog for IPv4 redirect messages.
- *value*: Configures the logging interval. The range is minimum 30 seconds to maximum 1800 seconds. The default interval is 60 seconds.

Step 3 (Optional) Use the **no ip redirect syslog** command to disable the syslog for excessive IPv4 redirect messages.

Example:

```
switch(config)# no ip redirect syslog
```

Verify the IPv4 Configuration

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

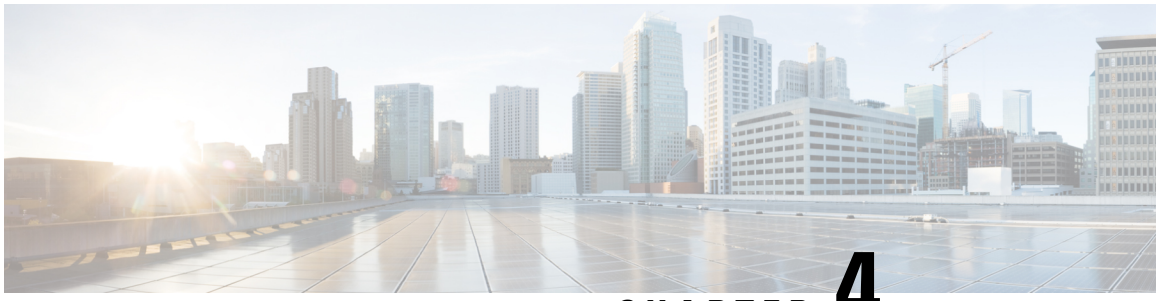
Command	Purpose
show ip adjacency	Displays the adjacency table.

Command	Purpose
show ip adjacency summary	Displays the summary of number of throttle adjacencies.
show ip arp	Displays the ARP table.
show ip arp summary	Displays the summary of the number of throttle adjacencies.
show ip interface	Displays IP-related interface information.
show ip arp statistics [**vrf** *vrf-name*]	Displays the ARP statistics.
show ip arp internal info interface <*interface-name*>	Displays the configured count and interval

Additional References

Related Documents for IPv4

Related Topic	Document Title
TCAM regions	See the Configuring ACL TCAM Region Sizes section in the Cisco Nexus 9000 Series NX-OS Security Configuration Guide.



CHAPTER 4

IPv6 Addresses

This chapter contains theses topics:

- IPv6 addresses, on page 57
- Virtualization support, on page 78
- IPv6 routes with ECMP, on page 79
- Prerequisites for IPv6, on page 79
- Guidelines and limitations for IPv6, on page 79
- Configure IPv6, on page 82
- Verify the IPv6 configuration, on page 100
- Verify the IPv6 configuration, on page 100

IPv6 addresses

IPv6 addresses are a network protocol designed to replace IPv4 addresses by expanding the address space and improving network functionality. IPv6 increases the address size from 32 bits in IPv4 to 128 bits, allowing for a vastly larger number of unique addresses. This expansion supports network scalability and global reachability.

Key attributes of IPv6 addresses include:

- A much larger address space that reduces the need for private addresses and Network Address Translation (NAT).
- A simplified main header and the use of extension headers, which improve packet processing efficiency.
- Support for new application protocols that operate without requiring special processing by border routers.
- Enhanced routing capabilities through features like prefix aggregation, simplified network renumbering, and site multihoming.
- Compatibility with routing protocols such as Routing Information Protocol (RIP), Integrated Intermediate System-to-Intermediate System (IS-IS), Open Shortest Path First (OSPF) for IPv6, and multiprotocol Border Gateway Protocol (BGP).

These attributes make IPv6 a more flexible and efficient protocol for modern and future network environments, enabling better routing, easier network management, and support for emerging applications.

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:BA98:7654:3210
```

```
2001:0DB8:0:0:8:800:200C: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 7: Compressed IPv6 Address Formats

IPv6 Address Type	Preferred Format	Compressed Format
Unicast	2001:0:0:0:0:DB8:800:200C: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 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 7.

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

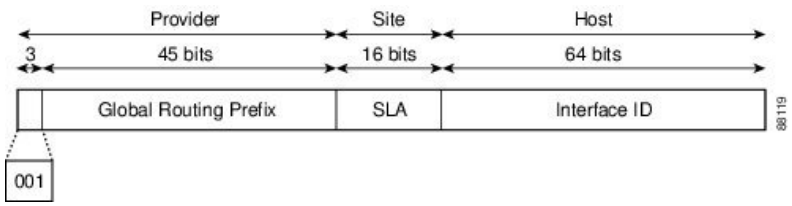
An aggregatable global address is an IPv6 address from the aggregatable global unicast prefix that

- enables strict aggregation of routing prefixes,
- limits the number of routing table entries in the global routing table, and
- is used on links aggregated upward through organizations to Internet service providers (ISPs).

This structure supports efficient routing by allowing hierarchical aggregation of addresses.

Aggregatable global IPv6 addresses are defined by a global routing prefix, a subnet ID, and an interface ID. All global unicast addresses, except those that start with binary 000, 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). See Figure 6: Aggregatable Global Address Format , on page 59 for 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 is a 16-bit subnet field that organizations can use to create a local addressing hierarchy and identify subnets. A subnet ID is similar to a subnet in IPv4. 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, and Frame Relay types), 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).

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.

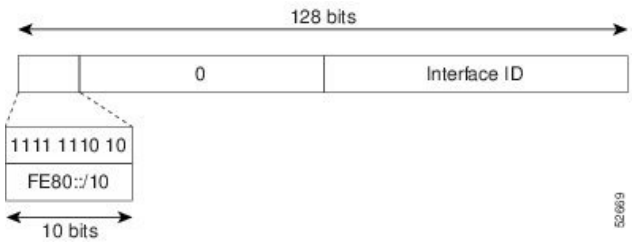
Link-Local addresses

A link-local address is an IPv6 unicast address. It 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) and the stateless autoconfiguration process. Nodes on a local link can use link-local addresses to communicate. These nodes do not need globally unique addresses to communicate. Figure Figure 6: Aggregatable Global Address Format , on page 59 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

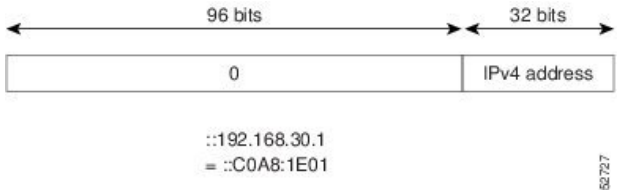


IPv4-Compatible IPv6 addresses

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. Figure Figure 6: Aggregatable Global Address Format , on page 59 shows the structure of a n 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.

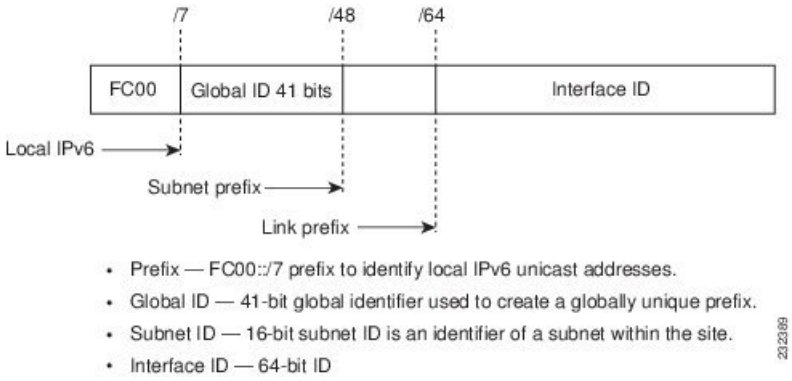
Unique local addresses are not expected to be routed on the global Internet. It can be routed inside of a limited area, such as a site, and it may be routed between a limited set of sites. Applications might treat unique local addresses like global scoped addresses.

These are the characteristics of unique local address:

- 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.

Figure Figure 6: Aggregatable Global Address Format , on page 59shows the structure of a unique local address.

Figure 9: Unique Local Address Structure



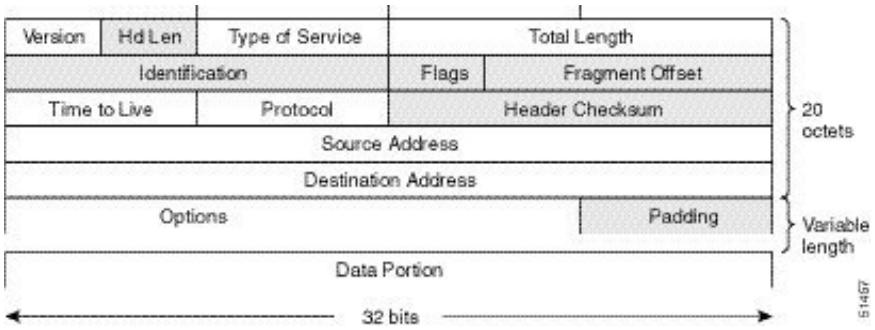
Site local addresses

Because RFC 3879 deprecates the use of site-local addresses, 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). 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 headers

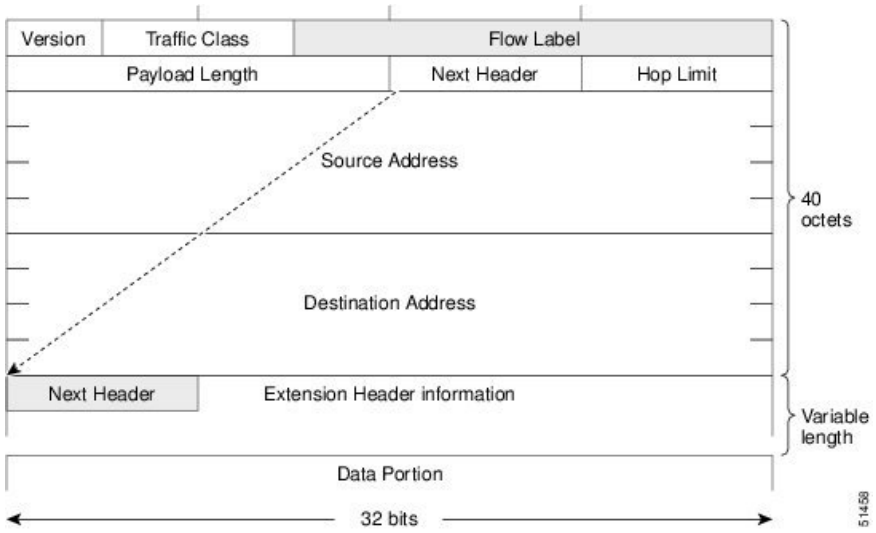
The base IPv6 packet header has 8 fields with a total size of 40 octets (320 bits). Fragmentation is handled by the source of a packet, and checksums at the data link layer and transport layer are used. The User Datagram Protocol (UDP) checksum checks the integrity of the inner packet, and the base IPv6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

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

Table 8: 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.
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 figure below.
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.

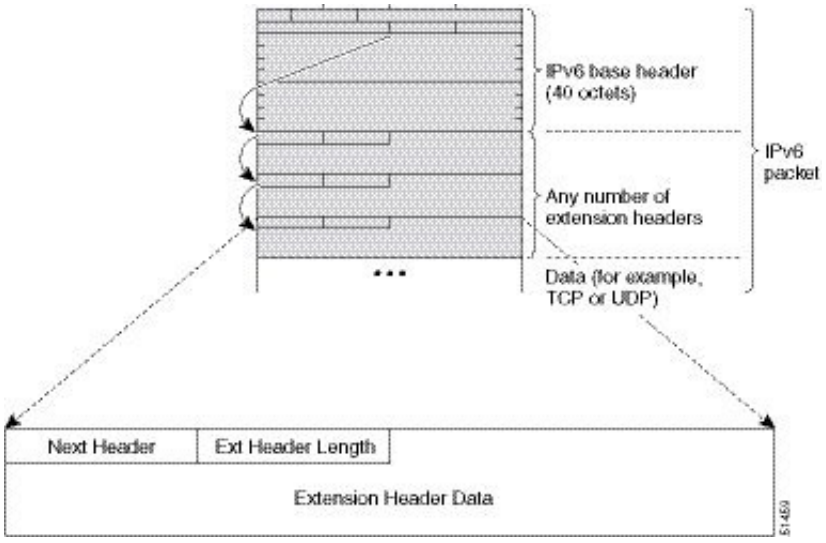
Figure 11: IPv6 Packet Header Format



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

Table 9: IPv6 Extension Header Types

Header Type	Next Header Value	Description
Hop-by-hop options	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	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.
Routing	43	Header that is used for source routing.
Fragment	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.
Authentication	51	Header that is used to provide connectionless integrity and data origin authentication for packets.
Encapsulation Security Payload	50	All information following this header is encrypted.
Mobility	135	Header that is used in support of Mobile IPv6 service.
Host Identity Protocol	139	Header that is used for Host Identity Protocol version 2 (HIPv2), which provides secure methods for IP multihoming and mobile computing.
Shim6	140	Header that is used for IP multihoming, which allows a host to be connected to multiple networks.
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).

Beginning with Cisco NX-OS Release 9.3(7), if you configure an IPv6 ACL on the devices listed here, you must include a new rule for the disposition of IPv6 packets that include extension headers. For the necessary configuration procedure, see "Configuring an ACL for IPv6 Extension Headers" in NX-OS Release 9.3(x) or later of the *Cisco Nexus 9000 Series NX-OS Security Configuration Guide*.

DNS for IPv6

IPv6 supports DNS record types that are supported in the DNS name-to-address and address-to-name lookup processes. See Table 10: IPv6 DNS Record Types, on page 66 table for the DNS record types support IPv6 addresses.

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

Table 10: 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 maximum transmission unit (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

Cisco Discovery Protocol (CDP) IPv6 address support is a feature that

- transfers IPv6 addressing information between directly connected Cisco devices,
- provides IPv6 neighbor information to network management and troubleshooting tools, and
- enhances network visibility and management for IPv6-enabled environments.

ICMP for IPv6

Internet Control Message Protocol for IPv6 (ICMPv6) is a network-layer protocol that

- works with IPv6
- reports errors if packets cannot be processed correctly, and
- sends informational messages about the status of the network.

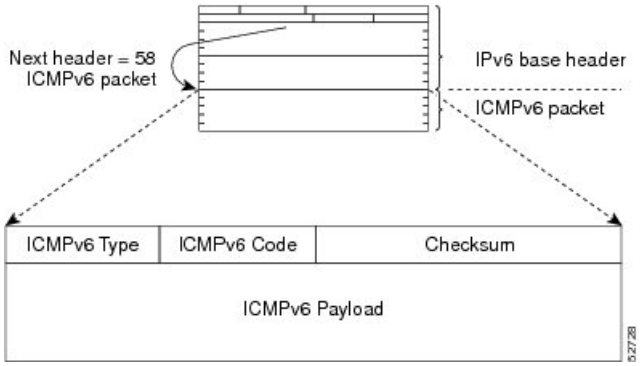
When a router is unable to forward a packet because it is too large to be sent out on another network, the router sends out an ICMPv6 message to the originating host. Additionally, ICMP packets in IPv6 are used in IPv6 neighbor discovery and path MTU discovery. The path MTU discovery process ensures 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. Figure Figure 13: IPv6 ICMP Packet Header Format , on page 68 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 NDP to

- determine the addresses of nodes on the same network (local link)
- find neighboring routers that can forward their packets
- verify whether neighboring routers are reachable or not, and
- to detect changes to link-layer addresses.

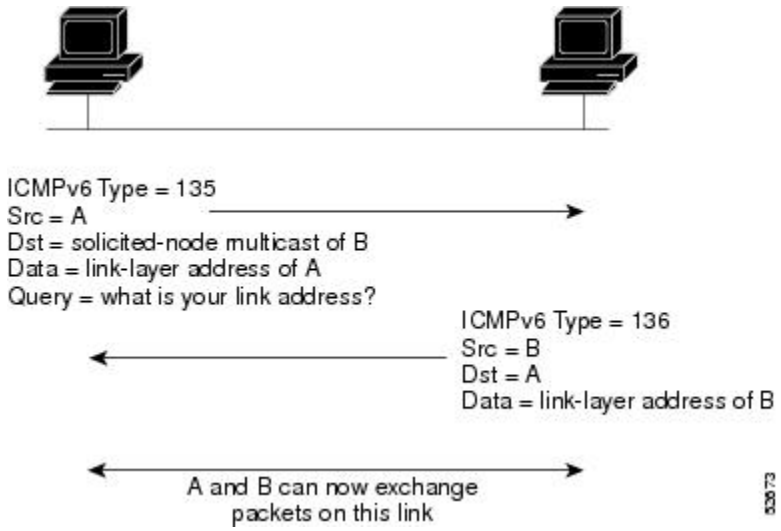
NDP uses ICMP messages to detect whether packets are sent to neighboring routers that are unreachable.

IPv6 neighbor solicitation messages

IPv6 neighbor solicitation messages are part of the Neighbor Discovery Protocol used by nodes on the same local link to determine each other's link-layer addresses and verify reachability.

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. 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. See Figure 14: IPv6 Neighbor Discovery-Neighbor Solicitation Message , on page 69 for more details.

Figure 14: IPv6 Neighbor Discovery-Neighbor Solicitation Message



A neighbor solicitation message is an ICMPv6 packet with a Type field value of 135. A node sends this message on the local link when it needs to find the link-layer address of another node on the same link. The message includes:

- The source address: the IPv6 address of the sending node.
- The destination address: the solicited-node multicast address corresponding to the target node's IPv6 address.
- The link-layer address of the source node.

When the destination node receives a neighbor solicitation, it replies with a neighbor advertisement message, which has a Type field value of 136. This message includes:

- The source address: the IPv6 address of the node sending the advertisement.
- The destination address: the IPv6 address of the node that sent the solicitation.
- The link-layer address of the advertising node.

After the source node receives this advertisement, the two nodes can communicate directly.

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.

Neighbor solicitation messages are also used in the stateless autoconfiguration process to verify the uniqueness of unicast IPv6 addresses before the addresses are assigned to an interface. Duplicate address detection is performed first on a new, link-local IPv6 address before the address is assigned to an interface (the new address remains in a tentative state while duplicate address detection is performed). A node sends a neighbor solicitation message with an unspecified source address and a tentative link-local address in the body of the message. If another node is already using that address, the node returns a neighbor advertisement message that contains the tentative link-local address. If another node is simultaneously verifying the uniqueness of the same address, that node also returns a neighbor solicitation message. If no neighbor advertisement messages are received in response to the neighbor solicitation message and no neighbor solicitation messages are received from other nodes that are attempting to verify the same tentative address, the node that sent the original neighbor solicitation message considers the tentative link-local address to be unique and assigns the address to the interface.

IPv6 stateless autoconfigurations

IPv6 Stateless Autoconfigurations are mechanisms that enable IPv6 nodes to automatically configure their own IP addresses without requiring manual setup or a DHCP server. These autoconfigurations ensure that every interface on an IPv6 node has a link-local address, which is essential for communication on the local network segment.

All interfaces on IPv6 nodes must have a link-local address, which is usually automatically configured from the identifier for an interface and the link-local prefix FE80::/10. A link-local address enables a node to communicate with other nodes on the link and can be used to further configure the node.

IPv6 Stateless Address Autoconfiguration (SLAAC) is performed only on a management interface. For example, when SLAAC is enabled on a management interface, it generates a Link Local Address (LLA) and performs a Duplicate Address Detection (DAD) on link local address. After the successful duplicate address detection process, the interface transmits ICMPv6 Router Solicitation (RS) packets. The upstream router that receives the RS packets responds back with an ICMPv6 Router Advertisement (RA). The RA packet will have a prefix TLV option that carries the subnet in which the downstream NX-OS Switch auto-generates the address, using the MAC information of the interface and the advertised prefix in RA packet. The Cisco NX-OS Switch auto-generates address in EUI-64 format and performs DAD on the new auto-generated addresses.

IPv6 addresses are assigned to an interface for a specific length of time. Each address has a lifetime that indicates how long the address is attached to an interface. The TLV prefix in the RA packet sent from the upstream router contain information about valid lifetime and preferred lifetime. The addresses that are assigned to an interface goes through two distinct phases. Initially, an address goes to a preferred state which means the address is not restricted for using in arbitrary communication. The address becomes deprecated state when the current interface binding becomes invalid. In a deprecated state, the use of the address is discouraged, not necessarily forbidden. Only applications that would have difficulty in switching to another address without a service disruption must use a deprecated address.

IPv6 compute node IP autoconfigurations

An IPv6 Compute Node IP Auto-Configuration is a feature that

- automatically assigns unique IPv6 node IP addresses to connected multi-homed compute nodes,
- enables these nodes to be onboarded into a Kubernetes (K8s) cluster, and
- establishes eBGP peering between the switch and each compute node.

A node IP must be assigned to connected compute nodes before they can be on-boarded into a K8s cluster and eBGP peering can be established between the switch and a compute node.

Beginning with Cisco NX-OS Release 10.3(3)F, the IPv6 Compute Node IP Auto-Configuration support is provided on Cisco NX-OS 9000 series platform switches to assign and distribute the node IP addresses to multi-homed compute nodes and establish reachability to K8s cluster using the assigned node IP.



Note The node address assignment is however different from SLAAC. It is a method to assign an unique IPv6 address on the loopback interface that is orthogonal to interface address provisioning in a layer-3 interface subnet that is done through SLAAC.

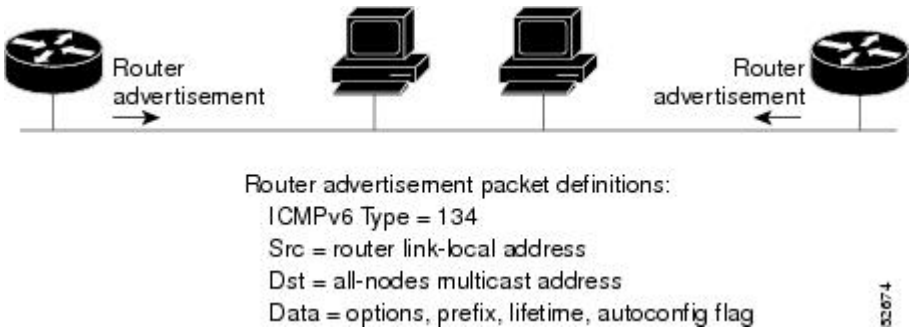
This feature complies with the standard as defined in RFC 8505/6775.

IPv6 router advertisement messages

IPv6 Router Advertisement (RA) messages are ICMP packets with a type field value of 134 that routers periodically send out on each configured interface. These messages enable IPv6 stateless6 stateless address autoconfiguration by advertising network prefixes and other configuration parameters to hosts on the local link.

The RA messages are sent to all-nodes multicast address as shown in the figure Figure 14: IPv6 Neighbor Discovery-Neighbor Solicitation Message , on page 69.

Figure 15: IPv6 Neighbor Discovery-RA Message



RA messages typically include these 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.
- Sets of flags that indicate the type of autoconfiguration (stateless or stateful) that can be completed.
- 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 these RA message parameters:

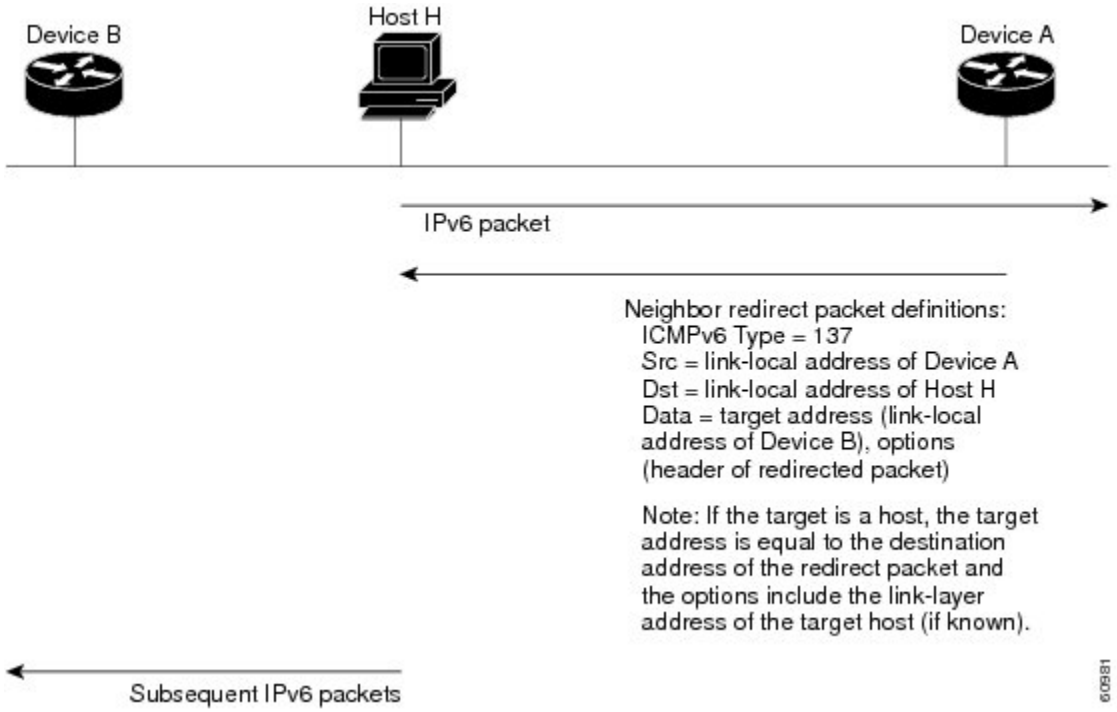
- The time interval between periodic RA messages.
- The router life-time value, which indicates the usefulness of a router as the default router (for use by all nodes on a given link).
- The network prefixes in use on a given link.
- The time interval between neighbor solicitation message retransmissions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

An IPv6 neighbor redirect message is a type of Internet Control Message Protocol (ICMP) packet that routers send to hosts to inform them of a better first-hop node on the path to a destination. This message is identified by a value of 137 in the Type field of the ICMP packet header.

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 theses 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.

IPv6 anycast addresses

An anycast address is an address that is assigned to a set of interfaces that belong to different nodes.

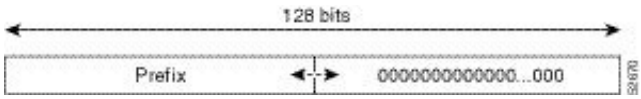
A packet sent to an anycast address is delivered to the closest interface as defined by the routing protocols in use identified by the anycast address. Anycast addresses are syntactically indistinguishable from unicast addresses because anycast addresses are allocated from the unicast address space.

Assigning a unicast address to more than one interface turns a unicast address into an anycast address. You must configure the nodes to which the anycast address belongs to recognize that the address is an anycast address.

Note Anycast addresses can be used only by a router, not by a host. Anycast addresses cannot be used as the source address of an IPv6 packet.

Figure Figure 17: Subnet Router Anycast Address Format , on page 74 represents the format of the subnet router anycast address. The address has a prefix concatenated by a series of zeros (the interface ID). The subnet router anycast address can be used to reach a router on the link that is identified by the prefix in the subnet router anycast address.

Figure 17: Subnet Router Anycast Address Format



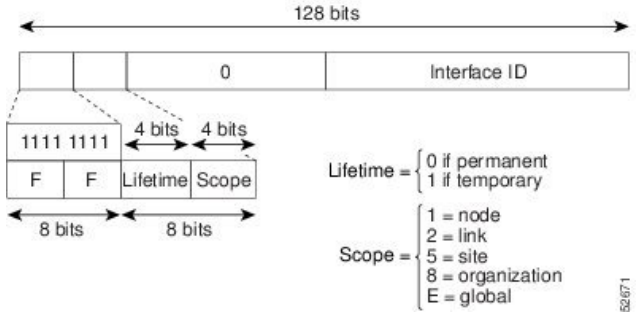
IPv6 multicast addresses

IPv6 multicast addresses are a type of IPv6 address that identify a group of interfaces across different nodes. These addresses always start with the prefix FF00::/8 (binary 1111 1111). When a packet is sent to an IPv6 multicast address, it is delivered to all interfaces that belong to the multicast group.

The second octet following the prefix defines the lifetime and scope of the multicast address. A permanent multicast address has a lifetime parameter equal to 0. A temporary multicast address has a lifetime parameter equal to 1.

A multicast address that has the scope of a node, link, site, or organization, or a global scope, has a scope parameter of 1, 2, 5, 8, or E, respectively. For example, a multicast address with the prefix FF02::/16 is a permanent multicast address with a link scope. Figure Figure 17: Subnet Router Anycast Address Format , on page 74 shows the format of the IPv6 multicast address.

Figure 18: IPv6 Multicast Address Format



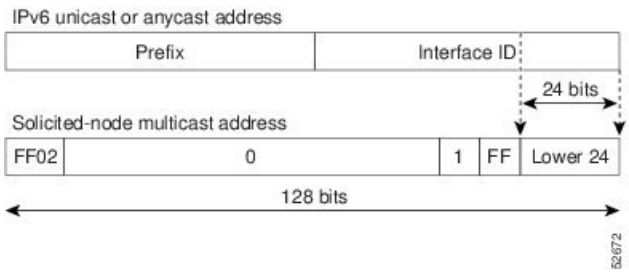
IPv6 nodes (hosts and routers) are required to join (where received packets are destined for) these multicast groups:

- All-nodes multicast group FF02:0:0:0:0:0:0:1 (the scope is link-local).
- Solicited-node multicast group FF02:0:0:0:0:1:FF00:0000/104 for each of its assigned unicast and anycast addresses.

IPv6 routers must also join the all-routers multicast group FF02:0:0:0:0:0:0:2 (the scope is link-local).

The solicited-node multicast address is a multicast group that corresponds to an IPv6 unicast or anycast address. IPv6 nodes must join the associated solicited-node multicast group for every unicast and anycast address to which they are assigned. The IPv6 solicited-node multicast address has the prefix FF02:0:0:0:0:1:FF00:0000/104 concatenated with the 24 low-order bits of a corresponding IPv6 unicast or anycast address (see the figure below). For example, the solicited-node multicast address that corresponds to the IPv6 address 2037::01:800:200E:8C6C is FF02::1:FF0E:8C6C. Solicited-node addresses are used in neighbor solicitation messages.

Figure 19: IPv6 Solicited-Node Multicast Address Format



Note IPv6 has no broadcast addresses. IPv6 multicast addresses are used instead of broadcast addresses.

LPM routing modes

By default, Cisco NX-OS programs routes in a hierarchical fashion to allow for the longest prefix match (LPM) on the device. However, you can configure the device for different routing modes to support more LPM route entries.

Theses tables list the LPM routing modes that are supported on Cisco Nexus 9000 Series switches.

LPM routing modes for Cisco Nexus 9200 platform switches

Table 11: LPM routing modes for Cisco Nexus 9200 platform switches

LPM routing mode	CLI command
Default system routing mode	
LPM dual-host routing mode	**system routing template-dual-stack-host-scale**
LPM heavy routing mode	**system routing template-lpm-heavy**

Note Cisco Nexus 9200 platform switches do not support the **system routing template-lpm-heavy** mode for IPv4 Multicast routes. Make sure to reset LPM's maximum limit to 0.

LPM routing modes for Cisco Nexus 9300 platform switches

Table 12: LPM routing modes for Cisco Nexus 9300 platform switches

LPM routing mode	Broadcom T2 mode	CLI command
Default system routing mode	3	
ALPM routing mode	4	**system routing max-mode l3**

LPM routing modes for Cisco Nexus 9300-EX/FX/FX2/FX3/GX platform switches

Table 13: LPM routing modes for Cisco Nexus 9300-EX/FX/FX2/FX3/GX platform switches

LPM routing mode	CLI command
LPM dual-host routing mode	**system routing template-dual-stack-host-scale**
LPM heavy routing mode	**system routing template-lpm-heavy**
LPM Internet-peering mode	**system routing template-internet-peering**

LPM routing modes for Cisco Nexus 9300-FX platform switches

LPM routing modes for Cisco Nexus 9300-fx2 platform switches

LPM routing modes for Cisco Nexus 9300-GX platform switches

LPM routing modes for Cisco Nexus 9500 platform switches with 9700-ex and 9700-fx line cards

Table 14: LPM routing modes for Cisco Nexus 9500 platform switches with 9700-ex and 9700-fx line cards

LPM routing mode	Broadcom T2 mode	CLI command
Default system routing mode	3 (for line cards); 4 (for fabric modules)	
Max-host routing mode	2 (for line cards); 3 (for fabric modules)	**system routing max-mode host**
Nonhierarchical routing mode	3 (for line cards); 4 with max-l3-mode option (for line cards)	**system routing non-hierarchical-routing [max-l3-mode]**
64-bit ALPM routing mode	Submode of mode 4 (for fabric modules)	**system routing mode hierarchical 64b-alpm**
LPM heavy routing mode		**system routing template-lpm-heavy** **Note** This mode is supported only for Cisco Nexus 9508 switches with the 9732C-EX line card.

LPM routing mode	Broadcom T2 mode	CLI command
LPM Internet-peering mode		**system routing template-internet-peering** **Note** This mode is supported only for the following Cisco Nexus 9500 Platform Switches: • Cisco Nexus 9500 platform switches with 9700-EX line cards. • Cisco Nexus 9500-FX platform switches (Cisco NX-OS release 7.0(3)I7(4) and later) • Cisco 9500-R platform switches (Cisco NX-OS release 9.3(1) and later)
LPM dual-host routing mode		

LPM routing modes for Cisco Nexus 9500-R platform switches with 9600-R line cards

Table 15: LPM Routing Modes for Cisco Nexus 9500-R Platform Switches with 9600-R Line Cards

LPM routing Mode	CLI command
LPM Internet-peering mode	**system routing template-internet-peering** (Cisco NX-OS release 9.3(1) and later)

Host to LPM spillover

Beginning with Cisco NX-OS Release 7.0(3)I5(1), host routes can be stored in the LPM table in order to achieve a larger host scale. In ALPM mode, the switch allows fewer host routes. If you add more host routes than the supported scale, the routes that are spilled over from the host table take the space of the LPM routes in the LPM table. The total number of LPM routes allowed in that mode is reduced by the number of host routes stored. This feature is supported on Cisco Nexus 9300 and 9500 platform switches.

In the default system routing mode, Cisco Nexus 9300 platform switches are configured for higher host scale and fewer LPM routes, and the LPM space can be used to store more host routes. For Cisco Nexus 9500 platform switches, only the default system routing and nonhierarchical routing modes support this feature on line cards. Fabric modules do not support this feature.

Virtualization support

IPv6 supports virtual routing and forwarding (VRF) instances.

IPv6 routes with ECMP

IPv6 routes with Equal-Cost Multipath (ECMP) are routing entries that distribute traffic across multiple next-hops with equal cost to optimize network performance and redundancy.

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.

Interface peering Backup paths is not supported for IPv4 address-families learnt over IPv6 neighbors.

Prerequisites for IPv6

These are the prerequisites of IPv6:

- You must be familiar with IPv6 basics such as IPv6 addressing and IPv6 header information.
- 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 these configuration guidelines and limitations:

- Cisco Nexus 9300-EX and Cisco Nexus 9300-FX2 platform switches configured for internet-peering mode might not have sufficient hardware capacity to install full IPv4 and IPv6 Internet routes simultaneously.
- 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.
- Usage of IPv6 LLA requires the TCAM Region for **ing-sup** to be re-carved from the default value of 512 to 768. This step requires a copy run start and reload
- IPv6 static route next-hop link-local addresses cannot be configured at any local interface.
- You must define the BGP update source when using a link-local IPv6 address.
- 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.
- When **system routing template-internet-peering** is configured, multicast traffic is not supported. However, it is still possible to have feature PIM and PIM configurations in the switch.

- For Cisco Nexus 9500-R platform switches, internet-peering mode is only intended to be used with the prefix pattern as distributed in the global internet routing table. In this mode, other prefix distributions/patterns can operate, but not predictably. As a result, maximum achievable LPM/LEM scale is reliable only when the prefix patterns are actual internet prefix patterns. In Internet-peering mode, if route prefix patterns other than those in the global internet routing table are used, the switch might not successfully achieve documented scalability numbers.
- LPM heavy routing mode is supported on Cisco Nexus 9500 series switches with 9700 -EX, -FX, and -GX series modules.
- Beginning with Cisco NX-OS Release 10.2(3)F, syslog will be printed when IPv6 redirect message is triggered based on the configured interval.
- Beginning with Cisco NX-OS Release 10.3(1)F, static routing is supported on the Cisco Nexus 9808 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, static routing is supported on the Cisco Nexus 9804 switches.
- Beginning with Cisco NX-OS Release 10.3(1)F, dynamic routing is supported on the Cisco Nexus 9808 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, dynamic routing is supported on the Cisco Nexus 9804 switches.
- Beginning with Cisco NX-OS Release 10.3(3)F, IPv6 Compute Node IP Auto-Configuration feature is supported on Cisco NX-OS 9000 series platform switches with the following limitations:
  - The RA prefix must be configured as offlink, with the prefix length of 64.
  - If there is a multi-homed compute node, same RA prefix must be configured on both L1 and L2 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, dynamic routing is supported on Cisco Nexus X98900CD-A and X9836DM-A line cards with 9808 and 9804 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, static routing is supported on Cisco Nexus X98900CD-A and X9836DM-A line cards with 9808 and 9804 switches.
- Beginning with Cisco NX-OS Release 10.4(2)F, static routing is supported on Cisco Nexus 9232E-B1 switches.
- Beginning with Cisco NX-OS Release 10.4(2)F, the following IPv6 Layer-3 features are supported on the Cisco Nexus 9232E-B1 switches:
  - IPv6 unicast routes
  - IPv6 host routes
  - IPv6 Neighbor Discovery
  - BGP Neighbors
- Beginning with Cisco NX-OS Release 10.6(1)F, the following IPv6 Layer-3 features are supported on the Cisco Nexus N9336C-SE1 switches:
  - IPv6 unicast routes

  - IPv6 host routes
  - IPv6 Neighbor Discovery
  - ARP

- Beginning with Cisco NX-OS Release 10.6(2)F, you can use **template-l3-host-heavy** to handle /128 host routes and increased IPv6 route scale up to the verified limit. This new template is introduced on Cisco Nexus N9300-FX2 platform switches.
- Beginning with Cisco NX-OS Release 10.6(2)F, IPv6 is supported on Cisco Nexus N9324C-SE1U and N9348Y2C6D-SE1U switches.
- Beginning with Cisco NX-OS Release 10.6(3)F, you can use **template-l3-host-heavy** on Cisco Nexus N9K-C9364C-H1, N9K-C9332D-H2R, and N9K-C93400LD-H1 platform switches.
- Beginning with Cisco NX-OS Release 10.6(2n), Layer 3 support for IPv6 is supported on the Cisco N9164E-NS4-O switch.
- Beginning with Cisco NX-OS Release 10.6(3)F, you can configure the neighbour discovery cache limit at the SVI level. This feature improves resource optimization, security, and network performance.
  - This feature enables the restriction on at the software level. The ND packets are still copied to slow path even after reaching limits.
  - You can enable this feature in interface mode as well as in global configuration mode. However, interface level configuration takes precedence over the global configuration mode.
  - This feature is supported on SVI and not supported on these interfaces:
    - Ethernet
    - Sub-interfaces
    - Port-channel
    - Un-numbered Interfaces
  - If users enable this feature in non-supporting interface mode such as management or ethernet interface, the configuration is applied in global configuration mode and this warning message is shown:

    ```
    Warning: ND cache limit configured globally. If ND cache count already exceeds the
    configured value, excessive entries will be removed in next refresh cycle.
    ```

  - Remove this configuration before downgrading to the releases earlier than NX-OS 10.6(3)F where this feature is not supported.

Configure IPv6

Configure IPv6 addresses

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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface ethernet** *number* command to enter interface configuration mode.

Example:

```
switch(config)# interface ethernet 2/3
switch(config-if)#
```

Step 3 Us the **ipv6 address** {*address* [**eui64**] [**route-preference** *preference*] [**secondary**] [**tag** *tag-id*] or **ipv6 address** *ipv6-address* **use-link-local-only** command to specify an IPv6 address assigned to the interface and enable IPv6 processing on the interface.

Example:

```
switch(config-if)# ipv6 address
2001:0DB8::1/10
```

or

```
switch(config-if)# ipv6 address
use-link-local-only
```

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.

Entering the **ipv6 address use-link-local-only** command configures a link-local address on the 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.

Step 4 (Optional) Use the **show ipv6 interface** command to see interfaces configured for IPv6.

Example:

```
switch(config-if)# show ipv6 interface
```

Step 5 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config
startup-config
```

Example

This example shows how to configure an IPv6 address:

```
switch# <userinput>configure terminal</userinput>
    switch(config)# <userinput>interface ethernet 3/1</userinput>
    switch(config-if)# <userinput>ipv6 address ?</userinput>
    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)# <userinput>ipv6 address 2001:db8::/64 eui64</userinput>
```

This example shows how to display an IPv6 interface:

```
switch(config-if)# <userinput>show ipv6 interface ethernet 3/1</userinput>
    Ethernet3/1, Interface status: protocol-down/link-down/admin-down, iod: 36
    IPv6 address: 2001:db8:0000:0000:0218:baff:fed8:239d
    IPv6 subnet: 2001:db8::/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
```

Configure max-host routing mode (Cisco Nexus 9500 Platform switches only)

By default, the device programs routes in a hierarchical fashion (with fabric modules that are configured to be in mode 4 and line card modules that are configured to be in mode 3), which allows for longest prefix match (LPM) and host scale on the device.

You can modify the default LPM and host scale to program more hosts in the system, as might be required when the node is positioned as a Layer-2 to Layer-3 boundary node.

Note If you want to further scale the entries in the LPM table, see the Configuring Nonhierarchical Routing Mode (Cisco Nexus 9500 Series Switches Only) section to configure the device to program all the Layer 3 IPv4 and IPv6 routes on the line cards and none of the routes on the fabric modules.

Note This configuration impacts both the IPv4 and IPv6 address families.

Note For the max-host routing mode scale numbers, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **system routing max-mode host** command to put the line cards in Broadcom T2 mode 2 and the fabric modules in Broadcom T2 mode 3 to increase the number of supported hosts.

Example:

```
switch(config)# system routing max-mode host
```

Step 3 (Optional) Use the **show forwarding route summary** command to see see the LPM routing mode.

Example:

```
switch(config)# show forwarding route summary
```

Step 4 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Step 5 Use the **reload** command to reboot the entire device.

Example:

```
switch(config)# reload
```

Configure nonhierarchical routing mode (Cisco Nexus 9500 series switches only)

If the host scale is small (as in a pure Layer 3 deployment), we recommend programming the longest prefix match (LPM) routes in the line cards to improve convergence performance. Doing so programs routes and hosts in the line cards and does not program any routes in the fabric modules.

Note This configuration impacts both the IPv4 and IPv6 address families.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **system routing non-hierarchical-routing** [**max-l3-mode**] to put the line cards in Broadcom T2 mode 3 (or Broadcom T2 mode 4 if you use the **max-l3-mode** option) to support a larger LPM scale.

Example:

```
switch(config)# system routing
 non-hierarchical-routing max-l3-mode
```

As a result, all of the IPv4 and IPv6 routes will be programmed on the line cards rather than on the fabric modules.

Step 3 (Optional) Use the **show forwarding route summary** command to see the LPM mode.

Example:

```
switch(config)# show forwarding route
                        summary
                        Mode 3:
                        120K IPv4 Host table
                        16k LPM table (> 65 < 127 1k entry
                        reserved)
                        Mode 4:
                        16k V4 host/4k V6 host
                        128k v4 LPM/20K V6 LPM
```

Step 4 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config
startup-config
```

Step 5 Use the **reload** command to reboot the entire device.

Example:

```
switch(config)# reload
```

Configure 64-Bit ALPM routing mode (Cisco Nexus 9500 platform switches only)

You can use the 64-bit algorithmic longest prefix match (ALPM) feature to manage IPv4 and IPv6 route table entries. In 64-bit ALPM routing mode, the device can store more route entries. In this mode, you can program one of the following:

- 80,000 IPv6 entries and no IPv4 entries
- No IPv6 entries and 128,000 IPv4 entries
- *x* IPv6 entries and y IPv4 entries, where $2x + y <= 128,000$

Note This configuration impacts both the IPv4 and IPv6 address families.

Note For the 64-bit ALPM routing mode scale numbers, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 use the [**no**] **system routing mode hierarchical 64b-alpm** command to program all IPv4 and IPv6 LPM routes with a mask length less than or equal to 64 in the fabric module.

Example:

```
switch(config)# system routing mode
hierarchical 64b-alpm
```

All host routes for IPv4 and IPv6 and all LPM routes with a mask length of 65–127 are programmed in the line card.

Step 3 (Optional) Use the **show forwarding route summary** command to see the LPM mode.

Example:

```
switch(config)# show forwarding route
summary
```

Step 4 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config
startup-config
```

Saves this configuration change.

Step 5 Use the **reload** command to reboot the entire device.

Example:

```
switch(config)# reload
```

Configure ALPM routing mode (Cisco Nexus 9300 platform switches only)

You can configure Cisco Nexus 9300 platform switches to support more LPM route entries.

This configuration impacts both the IPv4 and IPv6 address families.

- For ALPM routing mode scale numbers, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **system routing max-mode l3** command to put the device in Broadcom T2 mode 4 to support a larger LPM scale.

Example:

```
switch(config)# system routing max-mode l3
```

Step 3 (Optional) Use the **show forwarding route summary** command to see the LPM mode.

Example:

```
switch(config)# show forwarding
route summary
```

Step 4 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config
startup-config
```

Step 5 Use the **reload** command to reboot the entire device.

Example:

```
switch(config)# reload
```

Configure 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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface ethernet** *number* command to enter interface configuration mode.

Example:

```
switch(config)# interface ethernet 2/3
                        switch(config-if)#
```

Step 3 Use the **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**] command to specify an IPv6 address assigned to the interface and enables IPv6 processing on the interface.

Example:

```
switch(config-if)# ipv6 nd prefix
```

- **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**— Advertises the maximum transmission unit (MTU) in ICMPv6 router-advertisement messages on this link. The range is from 1280 to 65535 bytes.
- **ns-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**— Configures the interval between sending ICMPv6 router-advertisement messages. The range is from 4 to 1800 seconds.
- **ra-lifetime**— Advertises the lifetime of a default router in ICMPv6 router-advertisement messages. The range is from 0 to 9000 seconds.
- **reachable-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.



Note When disabling IPv6 redirects, IPv4 redirects should also be disabled as some IPv6 packets may still be leaked to the CPU.

- **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) Use the **show ip nd interface** command to see interfaces configured for IPv6 neighbor discovery.

Example:

```
switch(config-if)# show ip interface
```

Step 5 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config startup-config
```

Example

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

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

This example shows how to display an IPv6 interface:

```
switch# <userinput>configure terminal</userinput>
               switch(config)# <userinput>show ipv6 nd interface ethernet 3/1</userinput>
               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:

Table 16:

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.
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.

Configure IPv6 ND cache limit at the SVI

You can configure ND cache limit at the global configuration level as well as in interface level.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

```
switch# configure terminal

switch(config)#
```

-

Step 2 Use the **[no] ipv6 nd cache intf-limit** *limit* command to configure ND cache limit at the global configuration level.

Example:

```
switch(config)# ipv6 nd cache intf-limit 10

Range: 1-98000
```

Use the **no** form of this command to remove the configuration.

When you configure this CLI, ICMP generates this syslog warning message.

Warning ND cache limit configured globally. If ND cache count already exceeds the configured value, excessive entries will be removed in next refresh cycle.

-

Step 3 Use the **interface vlan** *vlan* command to enter SVI configuration mode.

Example:

```
switch(config)# interface vlan 10

switch(config-if)
```

-

Step 4 Use the **no] ipv6 nd cache intf-limit limit** *limit* command to configure ND cache limit at the SVI level.

Example:

```
switch(config-if)# ipv6 nd cache intf-limit 10

Range: 1-98000
```

Use the **no** form of this command to remove the configuration.

SVI level configuration takes precedence over the global configuration mode.

If the configured ND cache limit per interface is less than the currently existing total number of dynamically resolved ND entries for a given interface, ICMPv6 generates this syslog message:

```
Current number of ND cache entries(#total-count) exceeds the configured limit(#limit) for interface
(#intf-name) - Excessive entries will be removed in next refresh cycle.
```

After configuring, when ND cache count reaches the configured limit, ICMPv6 generates this syslog message:

```
Number of ND cache entries exceeded configured limit (#limit) for interface #Interface-name
```

-

Configure 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.

Procedure

Step 1 Enable IPv6 IDS checks in global configuration mode using the following commands as needed:

Example:

```
hardware ip verify address {destination zero | identical | reserved | source multicast}
                       hardware ipv6 verify length {consistent | maximum {max-frag | max-tcp | udp}}

                        hardware ipv6 verify tcp tiny-frag
                        hardware ipv6 verify version
```

The following commands perform various IDS checks on IPv6 packets:

Table 17: IPv6 IDS Command Reference

Command	Description
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).
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).

The IDS checks are enabled for IPv6 packet verification as configured.

Step 2 Display the IPv6 packet verification configuration.

Example:

```
show hardware forwarding ip verify
```

The current IPv6 packet verification configuration is displayed.

Configure IPv6 stateless autoconfiguration

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

```
Device# configure terminal
```

Step 2 Use the **interface management** *number* command to specify an interface type and number, and place the device in interface configuration mode.

Example:

```
switch(config)# interface mgmt0
```

Step 3 Use the **ipv6 address autoconfig** command to enable automatic configuration of IPv6 addresses using stateless autoconfiguration on the management interface.

Example:

```
switch(config-if)# ipv6 address autoconfig
```

Step 4 Use the **ipv6 address autoconfig default** command to enable automatic configuration of IPv6 addresses using stateless autoconfiguration on the management interface and adds a default route with next-hop as that of the link-local address received in the router advertisement.

Example:

```
switch(config-if)# ipv6 address autoconfig default
```

Example

This example shows how to use the show ipv6 interface command to display and verify that IPv6 addresses are configured on the management interface. Information displays the all the IPV6 addresses configured on the interface including the SLAAC generated addresses. It also indicates whether or not the stateless address autoconfig is enabled on the interface:

```
Device# show ipv6 interface mgmt 0

               IPv6 Interface Status for VRF "management"(2)
               mgmt0, Interface status: protocol-up/link-up/admin-up, iod: 2
               IPv6 address:
```

```
                1955::2f6:63ff:fe8b:c9f8/64 [VALID]
                IPv6 subnet: 1955::/64
                IPv6 link-local address: fe80::2f6:63ff:fe8b:c9f8 (default) [VALID]
                …..
                Stateless autoconfig configured on the interface


                This example shows how to use the show ipv6 route vrf management command
to display the IPv6 routing table for VRF management:

                Device# show ipv6 route vrf management

                IPv6 Routing Table for VRF "management"
                '*' denotes best ucast next-hop
                '**' denotes best mcast next-hop
                '[x/y]' denotes [preference/metric]
                0::/0, ubest/mbest: 1/0
                *via fe80::2f6:63ff:fe8b:c9ff, mgmt0, [2/0], 00:02:00, icmpv6
                1955::/64, ubest/mbest: 1/0, attached
                *via 1955::2f6:63ff:fe8b:c9f8, mgmt0, [0/0], 15:59:22, direct,
                1955::2f6:63ff:fe8b:c9f8/128, ubest/mbest: 1/0, attached
                *via 1955::2f6:63ff:fe8b:c9f8, mgmt0, [0/0], 15:59:22, local


                This example shows how to use the show ipv6 nd int mgmt command to display
 the ICMPv6 ND interfaces for VRF management:

                Device# show ipv6 nd int mgmt 0

                ICMPv6 ND Interfaces for VRF "management"
                mgmt0, Interface status: protocol-up/link-up/admin-up
                IPv6 address:
                1955::2f6:63ff:fe8b:c9f8/64 [VALID]
                IPv6 link-local address: fe80::2f6:63ff:fe8b:c9f8 [VALID]
                …….
                Subnets configured via SLAAC and their states:
                Prefix 1955::/64[PREFERRED] Preferred lifetime left: 6d23h Valid lifetime
lef
                t: 4w1d
```

Configure LPM heavy routing mode (Cisco Nexus 9200 and 9300-EX platform switches and 9732C-EX Line card only)

Beginning with Cisco NX-OS Release 7.0(3)I4(4), you can configure LPM heavy routing mode in order to support significantly more LPM route entries. Only the Cisco Nexus 9200 and 9300-EX Series switches and the Cisco Nexus 9508 switch with an 9732C-EX line card support this routing mode.

Note This configuration impacts both the IPv4 and IPv6 address families.

Note For LPM heavy routing mode scale numbers, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **system routing template-lpm-heavy** command to put the device in LPM heavy routing mode to support a larger LPM scale.

Example:

```
switch(config)# system routing template-lpm-heavy
```

Step 3 (Optional) Use the **show system routing mode** command to see the LPM routing mode.

Example:

```
switch(config)# show system routing mode
                        Configured System Routing Mode: LPM Heavy
                        Applied System Routing Mode: LPM Heavy
```

Step 4 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Step 5 Use the **reload** command to reboot the entire device.

Example:

```
switch(config)# reload
```

Configure L3 host heavy template

Beginning with Cisco NX-OS Release 10.6(2)F, you can use **template-l3-host-heavy** to handle /128 host routes and increased IPv6 route scale up to the verified limit. This new template is introduced on Cisco Nexus N9300-FX2 platform switches.

Beginning with Cisco NX-OS Release 10.6(3)F, you can use this template on Cisco Nexus N9K-C9364C-H1, N9K-C9332D-H2R, and N9K-C93400LD-H1 platform switches.

TCAM Tcam space carving does not come from the template changes. To achieve the required LPM in TCAM, apply the these configurations:

- **hardware profile multicast max-limit lpm-entries 0**
- **hardware profile ipv6 lpm-entries maximum 6144**

Procedure

Step 1 Use the configure terminal. command to enter global configuration mode.

Example:

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

Step 2 Use the [no] system routing template-l3-host-heavy command to configure L3 Host Heavy Template.

Example:

```
switch(config)# system routing template-l3-host-heavy
```

Configure LPM internet-peering routing mode (Cisco Nexus 9500-R platform switches, Cisco Nexus 9300-EX platform switches and Cisco Nexus 9000 series switches with 9700-EX line cards only)

Beginning with Cisco NX-OS Release 7.0(3)I6(1), you can configure LPM Internet-peering routing mode in order to support IPv4 and IPv6 LPM Internet route entries. This mode supports dynamic Trie (tree bit lookup) for IPv4 prefixes (with a prefix length up to /32) and IPv6 prefixes (with a prefix length up to /83). Only the Cisco Nexus 9300-EX platform switches and Cisco Nexus 9500 platform switches with 9700-EX line cards support this routing mode.

Beginning with Cisco NX-OS Release 9.3(1), Cisco Nexus 9500-R platform switches support this routing mode.

When **system routing template-internet-peering** is configured, multicast traffic is not supported. However, it is still possible to have feature PIM and PIM configurations in the switch.

Note This configuration impacts both the IPv4 and IPv6 address families.

Note For LPM Internet-peering routing mode scale numbers, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide. Cisco Nexus 9500-R platform switches in LPM Internet-peering mode scale out prectably only if they use internet-peering prefixes. If a Cisco Nexus 9500-R platform switch uses other prefix patterns, it might not achieve documented scalability numbers.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **system routing template-internet-peering** command to put the device in LPM Internet-peering routing mode to support IPv4 and IPv6 LPM Internet route entries.

Example:

```
switch(config)# system routing template-internet-peering
```

Step 3 (Optional) use the **show system routing mode** command to see see the LPM routing mode.

Example:

```
switch(config)# show system routing mode
                        Configured System Routing Mode: Internet Peering
                        Applied System Routing Mode: Internet Peering
```

Step 4 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Step 5 Use the **reload** command go reboot the entire device.

Example:

```
switch(config)# reload
```

Additional configuration for LPM internet-peering routing mode

When you deploy a Cisco Nexus switch in LPM Internet-peering routing mode in a large-scale routing environment or for routes with an increased number of next hops, you need to increase the memory limits for IPv4 under the VDC resource template.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 (Optional) Use the **show routing ipv4 memory estimate routes routes next-hops hops** command to see shared memory estimates to help you determine the memory requirements for routes.

Example:

```
switch(config)# show routing ipv4 memory estimate routes 262144 next-hops 32
                        Shared memory estimates:
                        Current max 512 MB; 78438 routes with 64 nhs
                        in-use 2 MB; 2642 routes with 1 nhs (average)
                        Configured max 512 MB; 78438 routes with 64 nhs
                        Estimate memory with fixed overhead: 1007 MB; 262144 routes with 32 nhs
                        Estimate with variable overhead included:
                        - With MVPN enabled VRF: 1136 MB
                        - With OSPF route (PE-CE protocol): 1375 MB
                        - With EIGRP route (PE-CE protocol): 1651 M
```

Step 3 Use the **vdc switch id id** command to specify the VDC switch ID.

Example:

```
switch(config)# vdc switch id 1
switch(config-vdc)#
```

Step 4 Use the **limit-resource u4route-mem minimum min-limit maximum max-limit** command to configure the limits for IPv4 memory in megabytes.

Example:

```
switch(config-vdc)# limit-resource u4route-mem minimum 1024 maximum 1024
```

Note Beginning with Cisco Nexus Release 10.2(2)F, this command is only applicable to the 32-bit version of the software.

Step 5 Use the **exit** command to exit the VDC configuration mode.

Example:

```
switch(config-vdc)# exit
switch(config)#
```

Step 6 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Step 7 Use the **reload** command to reboot the entire device.

Example:

```
switch(config)# reload
```

Configure LPM dual-host routing mode (Cisco Nexus 9200 and 9300-EX platform switches)

You can configure LPM heavy routing mode in order to support more LPM route entries. Only the Cisco Nexus 9200 and 9300-EX platform switches and the Cisco Nexus 9508 switch with a 9732C-EX line card support this routing mode.

Note This configuration impacts both the IPv4 and IPv6 address families.

Note For LPM heavy routing mode scale numbers, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **system routing template-lpm-heavy** command to put the device in LPM heavy routing mode to support a larger LPM scale.

Example:

```
switch(config)# system routing template-lpm-heavy
```

Step 3 (Optional) Use the **show system routing mode** command to see the LPM routing mode.

Example:

```
switch(config)# show system routing mode

                        Configured System Routing Mode: LPM Heavy

                        Applied System Routing Mode: LPM Heavy
```

Step 4 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Step 5 Use the **reload** command to reboot the entire device.

Example:

```
switch(config)# reload
```

Configure IPv6 redirect syslog

To enable/disable the IPv6 redirect syslog or change the logging interval, use the below CLIs:

Note By default, redirecting syslog will be enabled.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **ipv6 redirect syslog** [<*value*>] command to configure the syslog for excessive IPv6 redirect messages.

Example:

```
switch(config)# ip redirect syslog 60
                        switch(config)#
```

- **ipv6 redirect syslog**: Enables the syslog for IPv6 redirect messages.
- *value*: Configures the logging interval. The range is minimum 30 seconds to maximum 1800 seconds. The default interval is 60 seconds.

Step 3 (Optional) Use the **no ipv6 redirect syslog** command to disable the syslog for excessive IPv6 redirect messages.

Example:

```
switch(config)# no ipv6 redirect syslog
```

Verify the IPv6 configuration

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

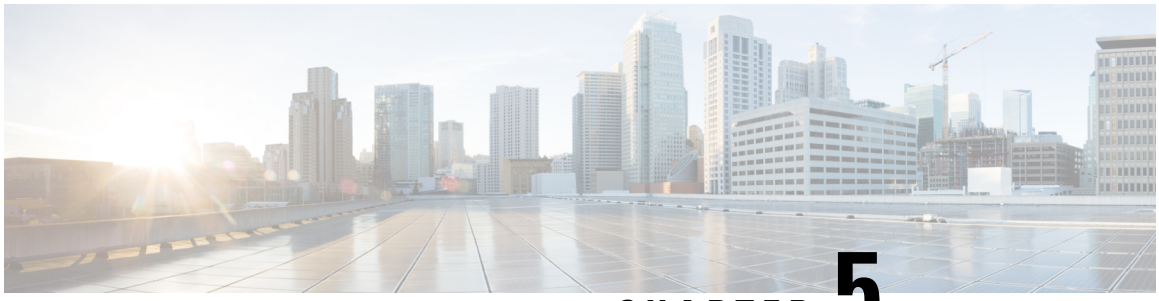
Command	Purpose
show hardware forwarding ip verify	Displays the IPv4 and IPv6 packet verification configuration.
show ipv6 interface	Displays IPv6-related interface information.
show ipv6 adjacency	Displays the adjacency table.
show system routing mode	Displays the LPM routing mode.
show ipv6 icmp	Displays ICMPv6 information.
show ipv6 nd	Displays IPv6 neighbor discovery interface information.
show ipv6 neighbor	Displays IPv6 neighbor entry.
show ipv6 nd addr-registry	Displays the IPv6 address registry entries of the compute node.

Verify the IPv6 configuration

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

Command	Purpose
show hardware forwarding ip verify	Displays the IPv4 and IPv6 packet verification configuration.

Command	Purpose
show ipv6 interface	Displays IPv6-related interface information.
show ipv6 adjacency	Displays the adjacency table.
show system routing mode	Displays the LPM routing mode.
show ipv6 icmp	Displays ICMPv6 information.
show ipv6 nd	Displays IPv6 neighbor discovery interface information.
show ipv6 neighbor	Displays IPv6 neighbor entry.
show ipv6 nd addr-registry	Displays the IPv6 address registry entries of the compute node.



CHAPTER 5

DNS

This chapter describes how to configure the Domain Name Server (DNS) client on the Cisco NX-OS device.

This chapter includes these sections:

- DNS clients, on page 103
- High availability, on page 104
- Virtualization support, on page 104
- Prerequisites for DNS clients, on page 104
- Guidelines and limitations for DNS clients, on page 104
- Default settings for DNS clients, on page 105
- Configure DNS clients, on page 105

DNS clients

DNS client

If your network devices require connectivity with devices in networks for which you do not control the name assignment, you can assign device names that uniquely identify your devices within the entire internetwork using the domain name server (DNS).

DNS uses a hierarchical scheme for establishing host names for network nodes, which allows local control of the segments of the network through a client-server scheme. The DNS system can locate a network device by translating the hostname of the device into its associated IP address.

On the Internet, a domain is a portion of the naming hierarchy tree that refers to general groupings of networks based on the organization type or geography. Domain names are pieced together with periods (.) as the delimiting characters. For example, Cisco is a commercial organization that the Internet identifies by a *com* domain, so its domain name is *cisco.com*. A specific hostname in this domain, the File Transfer Protocol (FTP) system, for example, is identified as *ftp.cisco.com*.

Name Servers

Name servers keep track of domain names and know the parts of the domain tree for which they have complete information. A name server may also store information about other parts of the domain tree. To map domain names to IP addresses in Cisco NX-OS, you must identify the hostnames, specify a name server, and enable the DNS service.

Cisco NX-OS allows you to statically map IP addresses to domain names. You can also configure Cisco NX-OS to use one or more domain name servers to find an IP address for a host name.

DNS operations

A name server handles client-issued queries to the DNS server for locally defined hosts within a particular zone by performing these actions:

- An authoritative name server responds to DNS user queries for a domain name that is under its zone of authority by using the permanent and cached entries in its own host table. If the query is for a domain name that is under its zone of authority but for which it does not have any configuration information, the authoritative name server replies that no such information exists.
- A name server that is not configured as the authoritative name server responds to DNS user queries by using information that it has cached from previously received query responses. If no router is configured as the authoritative name server for a zone, queries to the DNS server for locally defined hosts receive nonauthoritative responses.

Name servers answer DNS queries (forward incoming DNS queries or resolve internally generated DNS queries) according to the forwarding and lookup parameters configured for the specific domain.

High availability

Cisco NX-OS supports stateless restarts for the DNS client. After a reboot or supervisor switchover, Cisco NX-OS applies the running configuration.

Virtualization support

Cisco NX-OS supports multiple instances of the DNS clients that run on the same system. You can configure a DNS client. You can optionally have a different DNS client configuration in each virtual routing and forwarding (VRF) instance.

Prerequisites for DNS clients

To configure DNS client, you must have a DNS name server on your network.

Guidelines and limitations for DNS clients

The DNS client has theses configuration guidelines and limitations:

- You configure the DNS client in a specific VRF. If you do not specify a VRF, Cisco NX-OS uses the default VRF.
- Beginning with Cisco NX-OS Release 7.0(3)I5(1), DNS supports IPv6 addresses.
- The **source-interface** feature for **ip name-server** CLI is supported on all the DME enabled platforms. Prior to NX-OS Release 10.4(3)F, DME is not supported on C92348GC-X. Therefore, **source-interface**

option is not supported on this platform. However, **source-interface** option is supported on C92348GC-X, from NX-OS Release 10.4(3)F onwards.

- Beginning with Cisco NX-OS Release 10.6(2)F, Cisco Nexus 9000 Series switches support namespace-specific DNS resolution for third-party Linux applications running in containers.
  - DNS configuration for each VRF is now also written to a dedicated **resolv.conf** file under the **/etc/netns/<vrf>/** directory.
  - Applications running in a specific namespace can resolve DNS hosts using glibc APIs, with name server entries isolated to their namespace.
  - Namespace folders are deleted when the last nameserver entry is removed from that namespace.
- Users cannot configure or modify IP host for system internal hostnames.

Default settings for DNS clients

The table lists the default settings for DNS client parameters.

Default DNS client parameters

Parameters	Default
DNS client	Enabled

Configure DNS clients

Configure DNS client

You can configure the DNS client to use a DNS server on your network.

Before you begin

Ensure that you have a domain name server on your network.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **ip host** *name address1* [*address2... address6*] command to define up to six static hostname-to-address mappings in the hostname cache.

Example:

```
switch(config)# ip host cisco-rtp
192.0.2.1
```

The address can be either an IPv4 address or an IPv6 address.

Step 3 (Optional) Use the **ip domain-name** *name* [**use-vrf** *vrf-name*] command to define the default domain name that Cisco NX-OS uses to complete unqualified hostnames.

Example:

```
switch(config)# ip domain-name
myserver.com
```

You can optionally define a VRF that Cisco NX-OS uses to resolve this domain name if it cannot be resolved in the VRF that you configured this domain name under.

Cisco NX-OS appends the default domain name to any hostname that does not contain a complete domain name before starting a domain-name lookup.

Step 4 (Optional) Use the **ip domain-list** *name* [**use-vrf** *vrf-name*] command to define additional domain names that Cisco NX-OS can use to complete unqualified hostnames.

Example:

```
switch(config)# ip domain-list
mycompany.com
```

You can optionally define a VRF that Cisco NX-OS uses to resolve these domain names if they cannot be resolved in the VRF that you configured this domain name under.

Cisco NX-OS uses each entry in the domain list to append that domain name to any hostname that does not contain a complete domain name before starting a domain-name lookup. Cisco NX-OS continues this process for each entry in the domain list until it finds a match.

Step 5 (Optional) Use the **ip name-server** *address1* [*address2... address6*] [**source-interface** *interface-name*] [**use-vrf** *vrf-name*] command to define up to six name servers.

Example:

```
switch(config)# ip name-server
192.0.2.22
```

The address can be either an IPv4 address or an IPv6 address.

You can optionally define a VRF or source interface that Cisco NX-OS uses to reach this name server if it cannot be reached in the VRF that you configured this name server under.

Note Multiple DNS servers are for the case of unresponsive servers.

If the first DNS server in the list replies to the DNS query with a reject, the remaining DNS servers are not queried. If the first one doesn't respond, the next DNS server in list is queried.

Step 6 (Optional) Use the **ip domain-lookup** command to enable DNS-based address translation.

Example:

```
switch(config)# ip domain-lookup
```

This feature is enabled by default.

Step 7 (Optional) Use the **show hosts** command to see information about DNS.

Example:

```
switch(config)# show hosts
```

Step 8 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config
startup-config
```

Example

This example shows how to configure a default domain name and enable DNS lookup:

```
switch#<userinput> configure terminal</userinput>
    switch(config)# <userinput>ip domain-name cisco.com</userinput>
    switch(config)# <userinput>ip name-server 192.0.2.1 use-vrf management</userinput>
    switch(config)# <userinput>ip domain-lookup</userinput>
    switch(config)# <userinput>copy running-config startup-config</userinput>
```

Configure virtualization

You can configure a DNS client within a VRF. If you do not enter VRF configuration mode, your DNS client configuration applies to the default VRF.

You can optionally configure a DNS client to use a specified VRF other than the VRF under which you configured the DNS client as a backup VRF. For example, you can configure a DNS client in the Red VRF but use the Blue VRF to communicate with the DNS server if the server cannot be reached through the Red VRF.

Before you begin

Ensure that you have a domain name server on your network.

Procedure

Step 1 Use the **vrf context** *vrf-name* command to create a VRF and enters VRF configuration mode.

Example:

```
switch(config)# vrf context Red
      switch(config-vrf)#
```

Step 2 (Optional) Use the **ip domain-name** *name* [**use-vrf** *vrf-name*] command to define the default domain name server that Cisco NX-OS uses to complete unqualified hostnames.

Example:

```
switch(config-vrf)# ip domain-name
      myserver.com
```

You can optionally define a VRF that Cisco NX-OS uses to resolve this domain name server if it cannot be resolved in the VRF under which you configured this domain name.

Cisco NX-OS appends the default domain name to any hostname that does not contain a complete domain name before starting a domain-name lookup.

Step 3 (Optional) Use the **ip domain-list** *name* [**use-vrf** *vrf-name*] command to define additional domain name servers that Cisco NX-OS can use to complete unqualified hostnames.

Example:

```
switch(config-vrf)# ip domain-list
      mycompany.com
```

You can optionally define a VRF that Cisco NX-OS uses to resolve this domain name server if it cannot be resolved in the VRF under which you configured this domain name.

Cisco NX-OS uses each entry in the domain list to append that domain name to any hostname that does not contain a complete domain name before starting a domain-name lookup. Cisco NX-OS continues this process for each entry in the domain list until it finds a match.

Step 4 (Optional) Use the **ip name-server** *address1* [*address2... address6*] [**use-vrf** *vrf-name*] command to define up to six name servers.

Example:

```
switch(config-vrf)# ip name-server
      192.0.2.22
```

The address can be either an IPv4 address or an IPv6 address.

You can optionally define a VRF that Cisco NX-OS uses to reach this name server if it cannot be reached in the VRF that you configured this name server under.

Note Multiple DNS servers are for the case of unresponsive servers.

If the first DNS server in the list replies to the DNS query with a reject, the remaining DNS servers are not queried. If the first one doesn't respond, the next DNS server in list is queried.

Step 5 (Optional) use the **show hosts** command to see information about DNS.

Example:

```
switch(config-vrf)# show hosts
```

Step 6 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config
      startup-config
```

Example

This example shows how to configure a default domain and enable DNS lookup within a VRF:

```
switch# <userinput>configure terminal</userinput>
    switch(config)# <userinput>vrf context Red</userinput>
    switch(config-vrf)# <userinput>ip domain-name cisco.com</userinput>
    switch(config-vrf)# <userinput>ip name-server 192.0.2.1 use-vrf management</userinput>
    switch(config-vrf)# <userinput>copy running-config startup-config</userinput>
```

Verify DNS client configuration

To display the DNS client configuration, perform one of these tasks:

Command	Purpose
show hosts	Displays information about DNS.

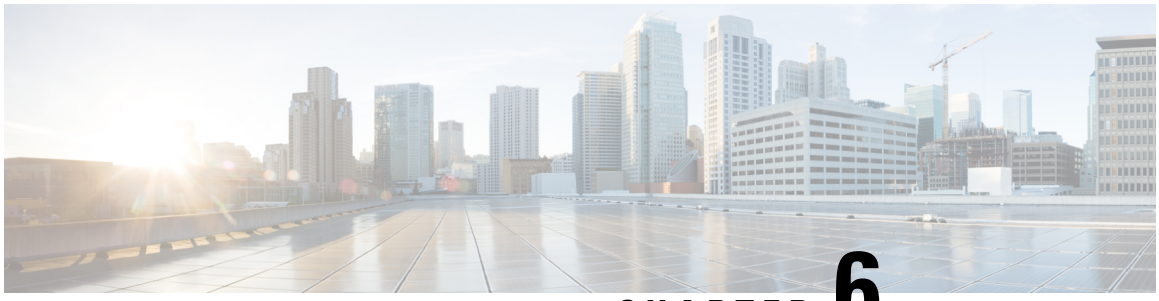
Configuration examples for the DNS client

This example shows how to establish a domain list with several alternate domain names:

```
ip domain-list csi.com
   ip domain-list telecomprog.edu
   ip domain-list merit.edu
```

This example shows how to configure the hostname-to-address mapping process and specify IP DNS-based translation. The example also shows how to configure the addresses of the name servers and the default domain name.

```
ip domain-lookup
   ip name-server 192.168.1.111 192.168.1.2
   ip domain-name cisco.com
```



CHAPTER 6

OSPFv2

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

This chapter includes the following sections:

- OSPFv2, on page 111
- OSPFv2 and the unicast RIB, on page 112
- Authentication, on page 112
- Advanced features, on page 113
- Prerequisites for OSPFv2, on page 118
- Guidelines and limitations for OSPFv2, on page 118
- Default settings for OSPFv2, on page 120
- Configure Basic OSPFv2, on page 121
- Configure Advanced OSPFv2, on page 131
- Verify the OSPFv2 configuration, on page 153
- Monitor OSPFv2, on page 155
- Configuration examples for OSPFv2, on page 155
- Related Documents for OSPFv2, on page 156
- Related Documents for OSPFv2, on page 156
- MIBs, on page 156

OSPFv2

OSPFv2 protocols are link-state are link-state routing protocols designed for IPv4 networks that enable routers to dynamically discover neighbors, establish adjacencies, and synchronize routing information. See Link-State Protocols to know more about link state protocols.

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 try 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. See Convergence to know more about network convergence.

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.

After the switch boots up, it sends OSPFv2 traps only after two dead intervals have passed to prevent flooding caused by frequent state changes.

OSPFv2 supports IPv4, while OSPFv3 supports IPv6. For more information, see OSPFv3, on page 157.



Note OSPFv2 on Cisco NX-OS supports RFC 2328. This RFC introduced a different method to calculate route summary costs which is not compatible with the calculation used by RFC1583. RFC 2328 also introduced different selection criteria for AS-external paths. It is important_ to ensure that all routers support the same RFC. RFC. Use the **rfc1583compatibility** command if your network includes routers that are only compliant with RFC1583. The default supported RFC standard for OSPFv2 may be different for Cisco NX-OS and Cisco IOS. You must make adjustments to set the values identically. See the OSPF RFC Compatibility Mode Example section for more information.

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 theses:

- Add or remove routes
- Handle route redistribution from other protocols
- Provide convergence updates to remove stale OSPFv2 routes and for stub router advertisements. For more information, see the SPFv2 Stub Router Advertisements.

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 clear-text password that is sent as part of the OSPFv2 message. The receiving OSPFv2 router must be configured with the same clear-text password to accept the OSPFv2 message as a valid route update.

Because the password is in clear text, anyone who can watch traffic on the network can learn the password.

Cryptographic authentication

Cryptographic authentication uses an encrypted password for OSPFv2 authentication. In this 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

MD5 authentication secures OSPFv2 messages by verifying their integrity and authenticity. It uses a shared password configured on 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.

HMAC-SHA authentication

Starting with Cisco NX-OS Release 7.0(3)I3(1), OSPFv2 supports RFC 5709 to allow the use of HMAC-SHA algorithms, which offer more security than MD5. The HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384. and HMAC-SHA-512 algorithms are supported for OSPFv2 authentication.

Advanced features

Cisco NX-OS supports advanced OSPFv3 features that enhance the usability and scalability of OSPFv2 in the network.

Stub areas

A stub area is an area that does not allow AS External (type 5) LSAs. These LSAs are usually flooded throughout the local autonomous system to propagate external route information. You can limit the amount

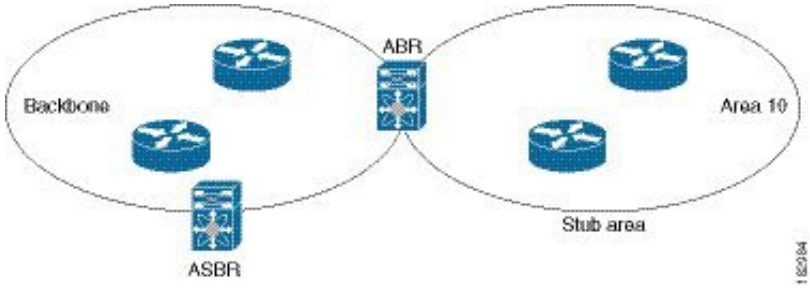
of external routing information that floods an area by making it a stub area. See Link-State advertisements, on page 161 to know more about LSA.

These are the requirements for stub area:

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

Figure Figure 20: Stub area, on page 114shows 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 20: 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 areas

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 ABR then floods these AS External LSAs throughout the OSPFv2 autonomous system. Summarization and filtering are supported during the translation. See Link-State advertisements, on page 161 for information about 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.

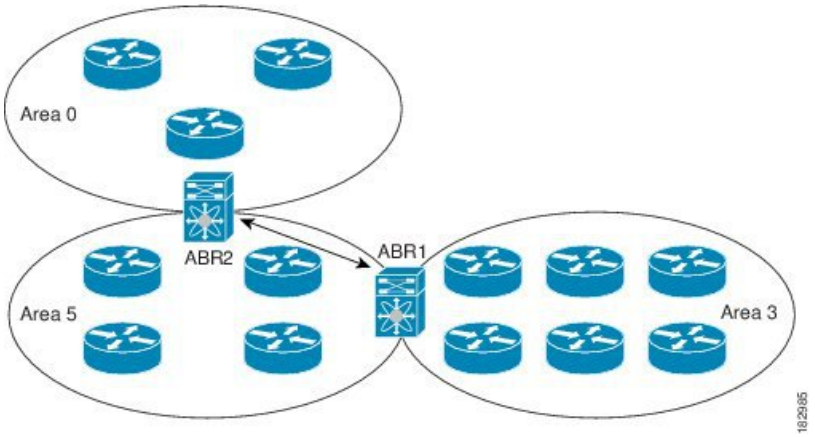
Note Beginning with Cisco NX-OS Release 9.2(4), OSPF became compliant with RFC 3101 section 2.5(3). When an Area Border Router attached to a Not-so-Stubby Area receives a default route LSA with P-bit clear, it should be ignored. OSPF had been previously adding the default route under these conditions.

If you have already designed your networks with RFC non-compliant behavior and expect a default route to be added on NSSA ABR, you will see a change in behavior when you upgrade to Cisco NX-OS Release 9.2(4) and later.

Virtual links

Virtual links allow you to connect an OSPFv2 area border router ABR to a backbone ABR when a direct physical connection is not available. The figure Figure 21: Virtual Links, on page 115 shows a virtual link that connects Area 3 to the backbone area through Area 5.

Figure 21: 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

Route redistribution is a processes that allows OSPFv2 to learn routes from other routing protocols. You can configure OSPFv2 to assign a specific link cost to these redistributed routes or apply a default link cost to all of them.

OSPFv2 can learn routes from other routing protocols by using route redistribution. See the Route Redistribution, on page 12. You can 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. 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 OSPFv2 autonomous system. See Configuring Route Policy Manager, for information about configuring route maps.

Route summarization

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.

You can use route summarization to reduce the number of unique routes that are flooded to every OSPF-enabled router because OSPFv2 shares all learned routes with every OSPF-enabled router.

Area border router (ABR) summarization is the process of aggregating routing information at the boundaries between OSPF areas. You typically perform summarization at ABRs to reduce the size of routing tables and improve network efficiency. Although you can configure summarization between any two areas, it is best to summarize routes toward the backbone area. This approach ensures that the backbone receives all summarized addresses and then injects these aggregates into other areas, optimizing routing updates and minimizing unnecessary detail across the network. There are two types of summarization:

- Inter-area route summarization
- External route summarization

Inter-area 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: It is specific to external routes that are injected into OSPFv2 using route redistribution. 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

High availabilities in Cisco NX-OS provide a multilevel architecture that ensures continuous network operation and minimal downtime. One key protocol supporting this is OSPFv2, which includes mechanisms for stateful restart and graceful restart to maintain routing stability during process interruptions.

OSPFv2 supports stateful restart, which is also referred to as non-stop routing (NSR). If OSPFv2 experiences problems, it attempts to restart from its previous run-time state. The neighbors do not register any neighbor event in this case. If the first restart is not successful and another problem occurs, OSPFv2 attempts a graceful restart.

A graceful restart, or nonstop forwarding (NSF), allows OSPFv2 to remain in the data forwarding path through a process restart. When OSPFv2 needs to perform a graceful restart, it sends a link-local opaque (type 9) LSA, called a grace LSA. This restarting OSPFv2 platform is called NSF capable.

The grace LSA includes a grace period, which is a specified time that the neighbor OSPFv2 interfaces hold onto the LSAs from the restarting OSPFv2 interface. (Typically, OSPFv2 tears down the adjacency and discards all LSAs from a down or restarting OSPFv2 interface.) The participating neighbors, which are called NSF helpers, keep all LSAs that originate from the restarting OSPFv2 interface as if the interface was still adjacent.

When the restarting OSPFv2 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 the graceful restart has finished.

Stateful restart is used in these scenarios:

- First recovery attempt after the process experiences problems
- User-initiated switchover using the **system switchover** command

Graceful restart is used in these scenarios:

- Second recovery attempt after the process experiences problems within a 4-minute interval.
- Manual restart of the process using the **restart ospf** command
- Active supervisor removal
- Active supervisor reload using the **reload module** *active-sup* command

OSPFv2 stub router advertisements

An OSPFv2 stub router advertisement is a feature that allows you to configure an OSPFv2 interface to act as a stub router. This feature helps you control OSPFv2 traffic through the router by limiting the routing load or managing network introduction of new routers.

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. For the number of supported OSPFv2 instances, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

SPF Optimization

The shortest path first (SPF) algorithm optimization in Cisco NX-OS is a set of techniques that

- improve the efficiency of SPF calculations,
- reduce CPU load during routing updates, and
- speed up network convergence.

Cisco NX-OS optimizes the SPF algorithm in these 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

Bidirectional forwarding detection (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.

OSFPv2 supports BFD. See the Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide for more information.

Virtualization support for OSPFv2

Cisco NX-OS supports multiple process instances for OSPFv2. Each OSPF instance can support multiple virtual routing and forwarding (VRF) instances, up to the system limit. For the number of supported OSPFv2 instances, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

Prerequisites for OSPFv2

These are the prerequisites for OSPFv2:

- 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 can communicate with a remote OSPFv2 neighbor.
- 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 Enabling OSPFv2 for more information.

Guidelines and limitations for OSPFv2

OSPFv2 has the following configuration guidelines and limitations:

- The **graceful-restart planned-only** command under OSPFv2 on **reload** converts to the **graceful-restart** command.

  This is not causing any impact on the functionality. If the **graceful-restart planned-only** is not in the configuration, this problem is not applicable for that device.

This occurs when the Cisco NX-OS release is 9.3(2) and CSCvs57583 is not included in the release. A workaround is to unconfigure the **graceful-restart** command and reconfigure the old command.

- Names in the prefix-list are case-insensitive. We recommend using unique names. Do not use the same name by modifying uppercase and lowercase characters. For example, CTCPrimaryNetworks and CtcPrimaryNetworks are not two different entries.
- If you enter the **no graceful-restart planned only** command, graceful restart is disabled.
- Cisco NX-OS displays areas in dotted decimal notation regardless of whether you enter the area in decimal or dotted decimal notation.
- All OSPFv2 routers must operate in the same RFC compatibility mode. OSPFv2 for Cisco NX-OS complies with RFC 2328. Use the **rfc1583compatibility** command in router configuration mode if your network includes routers that support only RFC 1583.
- In scaled scenarios, when the number of interfaces and link-state advertisements in an OSPF process is large, the snmp-walk on OSPF MIB objects is expected to time out with a small-values timeout at the SNMP agent. If your observe a timeout on the querying SNMP agent while polling OSPF MIB objects, increase the timeout value on the polling SNMP agent.
- The following guidelines and limitations apply to the administrative distance feature:
  - When an OSPF route has two or more equal cost paths, configuring the administrative distance is non-deterministic for the **match ip route-source** command.
  - Configuring the administrative distance is supported only for the **match route-type** , **match ip address prefix-list** , and **match ip route-source prefix-list** commands. The other match statements are ignored.
  - There is no preference among the **match route-type** , **match ip address** , and **match ip route-source** commands for setting the administrative distance of OSPF routes. In this way, the behavior of the table map for setting the administrative distance in Cisco NX-OS OSPF is different from that in Cisco IOS OSPF.
  - The discard route is always assigned an administrative distance of 220. No configuration in the table map applies to OSPF discard routes.
- If you configure the **delay restore** *seconds* command in vPC configuration mode and if the VLANs on the multichassis EtherChannel trunk (MCT) are announced by OSPFv2 or OSPFv3 using switch virtual interfaces (SVIs), those SVIs are announced with MAX_LINK_COST on the vPC secondary node during the configured time. As a result, all route or host programming completes after the vPC synchronization operation (on a peer reload of the secondary vPC node) before attracting traffic. This behavior allows for minimal packet loss for any north-to-south traffic.
- For N9K-X9636C-R and N9K-X9636Q-R line cards and the N9K-C9508-FM-R fabric module, the output of the **show run ospf** command might show the default values for some OSPF commands.



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.

- If you use the **network** *ip address mask* command under OSPF, an error message will be displayed, and you will be prompted to enable OSPF under an interface with **area** *area id* command.
- It is recommended that you use the OSPF default timers (hello-interval:10 and dead-interval:40). For better convergence time, you can use the BFD along with OSPF. This combination will give sub-second link/adjacency flaps detection and very low convergence time.
- While OSPF support are aggressive timers, these are not commended as aggressive timers will bring the adjancency down quickly as well as cause CPU churn. We recommend you to use the default timers and use BFD (Bidirectional Forwarding Detection) to get sub-second failure detection.
- Beginning with Cisco NX-OS Release 10.3(1)F, OSPFv2 is supported on the Cisco Nexus 9808 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, OSPFv2 is supported on the Cisco Nexus 9804 switches.
- Beginning with Cisco NX-OS Release 10.3(3)F, OSPFv2 supports Type-6 keychain encryption for OSPFv2 user password on the Cisco NX-OS switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, OSPFv2 is supported on Cisco Nexus X98900CD-A and X9836DM-A line cards with Cisco Nexus 9808 and 9804 switches.
- Beginning with Cisco NX-OS Release 10.4(2)F, OSPFv2 is supported on Cisco Nexus 9232E-B1 switches.
- Beginning with Cisco NX-OS Release 10.6(1)F, OSPFv2 is supported on Cisco Nexus N9336C-SE1 switches.
- Beginning with Cisco NX-OS Release 10.6(2)F, BFD (Single Hop) and OSPFv2 are supported on Cisco Nexus N9324C-SE1U and N9348Y2C6D-SE1U platform switches.

Default settings for OSPFv2

The table lists the default settings for OSPFv2 parameters.

Table 18: Default OSPFv2 parameters

Parameters	Default
Administrative distance	110
Hello interval	10 seconds
Dead interval	40 seconds
Discard routes	Enabled
Graceful restart grace period	60 seconds
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

Parameters	Default
LSA group pacing	10 seconds
SPF calculation initial delay time	200 milliseconds
SPF minimum hold time	5000 milliseconds
SPF calculation initial delay time	1000 milliseconds

Configure Basic OSPFv2

Configure OSPFv2 after you have designed your OSPFv2 network.

Enable OSPFv2

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

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **feature ospf** command to enable the OSPFv2 feature.

Example:

```
switch(config)# feature ospf
```

Step 3 (Optional) Use the **show feature** command to see enabled and disabled features.

Example:

```
switch(config)# show feature
```

Step 4 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

To disable the OSPFv2 feature and remove all associated configuration, use the no feature ospf command in global configuration mode:

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

Create 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 Configure Advanced OSPFv2, on page 131 section.

Before you begin

Ensure that you have enabled the OSPF feature. See the Enabling OSPFv2 for more information.

- 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

Step 1 Use the [**no**]**router ospf***instance-tag* command to create a new OSPFv2 instance with the configured instance tag.

Example:

```
switch(config)# router ospf 201
switch(config-router)
```

Step 2 (Optional) Use the **router-id***ip-address* command to configure the OSPFv2 router ID.

Example:

```
switch(config-router)# router-id 192.0.2.1
```

This IP address identifies this OSPFv2 instance and must exist on a configured interface in the system.

Step 3 (Optional) Use the **show ip ospf***instance-tag* command to see OSPF information.

Example:

```
switch(config-router)# show ip ospf 201
```

Step 4 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-router)# copy running-config
startup-config
```

Example

To remove the OSPFv2 instance and all associated configuration, use the no router ospf command in global configuration mode.

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



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

Configure Optional Parameters on an OSPFv2 Instance

You can configure optional parameters for OSPF, see the Configure Advanced OSPFv2, on page 131 section.

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

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

Step 1 Use the **distance** *number* command to configures the administrative distance for this OSPFv2 instance.

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 Use the **log-adjacency-changes [detail]** command to generates a system message whenever a neighbor changes state.

Example:

```
switch(config-router)#
log-adjacency-changes
```

Step 3 Use the **maximum-paths** *path-number* command configures the maximum number of equal OSPFv2 paths to a destination in the route table.

Example:

```
switch(config-router)# maximum-paths 4
```

This command is used for load balancing. The range is from 1 to 16. The default is 8.

Step 4 Use the **distance** *number* command to configure the administrative distance for this OSPFv2 instance.

Example:

```
switch(config-router)# distance 25
```

The range is from 1 to 255. The default is 110.

Step 5 Use the **log-adjacency-changes** [**detail**] command to generate a system message whenever a neighbor changes state.

Example:

```
switch(config-router)#
log-adjacency-changes
```

Step 6 Use the **maximum-paths** *path-number* command to configure the maximum number of equal OSPFv2 paths to a destination in the route table.

Example:

```
switch(config-router)# maximum-paths 4
```

This command is used for load balancing. The range is from 1 to 16. The default is 8.

Step 7 Use the **passive-interface default** command to suppress routing updates on all interfaces.

Example:

```
switch(config-router)# passive-interface
default
```

This command is overridden by the VRF or interface command mode configuration.

Step 8 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-router)# copy running-config
startup-config
```

Example

This example shows how to create an OSPFv2 instance:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router ospf 201</userinput>
switch(config-router)# <userinput>copy running-config startup-config</userinput>
```

Configure 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.

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

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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface***interface-type slot/port* command to enter interface configuration mode.

Example:

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

Step 3 Use the **ip address***ip-prefix/length* command to assign an IP address and subnet mask to this interface.

Example:

```
switch(config-if)# ip address
192.0.2.1/16
```

Step 4 Use the command **ip router ospf***instance-tag***area***area-id* [**secondaries none**] command to add the interface to the OSPFv2 instance and area.

Example:

```
switch(config-if)# ip router ospf 201
area 0.0.0.15
```

Step 5 (Optional) Use the **show ip ospf***instance-tag***interface***interface-type slot/port* command to see OSPF information.

Example:

```
switch(config-if)# show ip ospf 201
interface ethernet 1/2
```

Step 6 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config
startup-config
```

Step 7 (Optional) Use the **ip ospf cost***number* command to configure the OSPFv2 cost metric for this interface.

Example:

```
switch(config-if)# ip ospf cost 25
```

The default is to calculate cost metric, based on reference bandwidth and interface bandwidth. The range is from 1 to 65535.

Step 8 (Optional) Use the **ip ospf dead-interval***seconds* command to configure the OSPFv2 dead interval, in seconds.

Example:

```
switch(config-if)# ip ospf dead-interval 50
```

The range is from 1 to 65535. The default is four times the hello interval, in seconds.

Step 9 (Optional) Use the **ip ospf hello-interval***seconds* command to configure the OSPFv2 hello interval, in seconds.

Example:

```
switch(config-if)# ip ospf hello-interval
25
```

The range is from 1 to 65535. The default is 10 seconds.

Step 10 (Optional) Use the **ip ospf mtu-ignore** command to configure OSPFv2 to ignore any IP MTU mismatch with a neighbor.

Example:

```
switch(config-if)# ip ospf mtu-ignore
```

The default is to not establish adjacency if the neighbor MTU does not match the local interface MTU.

Step 11 (Optional) Use the **[default | no]ip ospf passive-interface** command to suppress routing updates on the interface.

Example:

```
switch(config-if)# ip ospf
passive-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.

Step 12 (Optional) Use the **ip ospf priority***number* command to configure the OSPFv2 priority, used to determine the DR for an area.

Example:

```
switch(config-if)# ip ospf priority 25
```

The range is from 0 to 255. The default is 1. See the Designated routers, on page 160 section.

Step 13 (Optional) Use the **ip ospf shutdown** command to shut down the OSPFv2 instance on this interface.

Example:

```
switch(config-if)# ip ospf shutdown
```

Example

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

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>interface ethernet 1/2</userinput>
switch(config-if)# <userinput>ip address 192.0.2.1/16</userinput>
switch(config-if)# <userinput>ip router ospf 201 area 0.0.0.10</userinput>
switch(config-if)# <userinput>copy running-config startup-config</userinput>
```

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.

Configure 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 key chain for this authentication configuration. See the Cisco Nexus 9000 Series NX-OS Security Configuration Guide.



Note For OSPFv2, the key identifier in the **key** *key-id* command supports values from 0 to 255 only.

Procedure

Step 1 Use the **router ospf***instance-tag* command to create a new OSPFv2 instance with the configured instance tag.

Example:

```
switch(config)# router ospf 201
switch(config-router)#
```

Step 2 Use the **area***area-id***authentication** [**message-digest**] command to configure the authentication mode for an area.

Example:

```
switch(config-router)# area 0.0.0.10
authentication
```

Step 3 Use the **interface***interface-type slot/port* command to enter interface configuration mode.

Example:

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

Step 4 (Optional) Use the **ip ospf authentication-key** [**0** | **3**]*key* command to configure simple password authentication for this interface.

Example:

```
switch(config-if)# ip ospf
authentication-key 0 mypass
```

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 5 (Optional) Use the **ip ospf message-digest-key***key-id***md5**[**0** | **3**]*key* command to configure message digest authentication for this interface.

Example:

```
switch(config-if)# ip ospf
message-digest-key 21 md5 0 mypass
```

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 clear text and 3 configures the pass key as 3DES encrypted.

Step 6 (Optional) Use the **show ip ospf***instance-tag***interface***interface-type slot/port* command to see OSPF information.

Example:

```
switch(config-if)# show ip ospf 201
interface ethernet 1/2
```

Step 7 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config
startup-config
```

Configure authentication for an interface

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 key chain for this authentication configuration. See the Cisco Nexus 9000 Series NX-OS Security Configuration Guide.

Note For OSPFv2, the key identifier in the **key** *key-id* command supports values from 0 to 255 only.

In Keychain, only key 0-255 will be supported by OSPFv2.

Procedure

Step 1 Use the **interface** *interface-type slot/port* command to enter interface configuration mode.

Example:

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

Step 2 Use the **ip ospf authentication** [**message-digest**] command to enable interface authentication mode for OSPFv2 for either cleartext or message-digest type.

Example:

```
switch(config-if)# ip ospf
authentication
```

Overrides area-based authentication for this interface. All neighbors must share this authentication type.

Step 3 (Optional) Use the **ip ospf authentication key-chain** *key-id* command to configure interface authentication to use key chains for OSPFv2.

Example:

```
switch(config-if)# ip ospf
authentication key-chain Test1
```

See the *Cisco NX-OS Series NX-OS Security Configuration Guide*, for details on key chains.

Step 4 (Optional) Use the **ip ospf authentication-key** [**0** | **3** | **7**] *key* command to configure simple password authentication for this interface.

Example:

```
switch(config-if)# ip ospf
authentication-key 0 mypass
```

Use this command if the authentication is not set to key-chain or message-digest.

The options are as follows:

- 0—Configures the password in clear text.
- 3—Configures the pass key as 3DES encrypted.
- 7—Configures the key as Cisco type 7 encrypted.

Step 5 (Optional) Use the **ip ospf message-digest-key** *key-id* **md5** [**0** | **3** | **7**] *key* command to configure message digest authentication for this interface.

Example:

```
switch(config-if)# ip ospf
message-digest-key 21 md5 0 mypass
```

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 clear text.
- 3—Configures the pass key as 3DES encrypted.
- 7—Configures the key as Cisco type 7 encrypted.

Step 6 (Optional) Use the **show ip ospf** *instance-tag* **interface** *interface-type slot/port* command to see OSPF information.

Example:

```
switch(config-if)# show ip ospf 201
interface ethernet 1/2
```

Step 7 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config
startup-config
```

Example

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

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router ospf 201</userinput>
switch(config-router)# <userinput>exit</userinput>
switch(config)# <userinput>interface ethernet 1/2</userinput>
switch(config-if)# <userinput>ip router ospf 201 area 0.0.0.10</userinput>
switch(config-if)# <userinput>ip ospf authentication</userinput>
switch(config-if)# <userinput>ip ospf authentication-key 0 mypass</userinput>
switch(config-if)# <userinput>copy running-config startup-config</userinput>
```

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 MD
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
```

Configure Advanced OSPFv2

Configure OSPFv2 after you have designed your OSPFv2 network.

Configure 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 as well, through an autonomous system border router (ASBR). See Areas, on page 160.

ABRs have the following optional configuration parameters:

- Area range—Configures route summarization between areas. See the Configuring Route Summarization.
- Filter list—Filters the Network Summary (type 3) LSAs 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.

- 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. See Areas, on page 160.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **router ospf** *instance-tag* command to create a new OSPFv2 instance with the configured instance tag.

Example:

```
switch(config)# router ospf 201
switch(config-router)#
```

Step 3 Use the **area** *area-id* **filter-list route-map** *map-name* {**in** | **out**} command to filter incoming or outgoing Network Summary (type 3) LSAs on an ABR.

Example:

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

Step 4 (Optional) Use the **show ip ospf policy statistics area** *id* **filter-list** {**in** | **out**} command to see OSPF policy information.

Example:

```
switch(config-router)# show ip ospf policy
statistics area 0.0.0.10 filter-list in
```

Step 5 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

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

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router ospf 201</userinput>
switch(config-router)# <userinput>area 0.0.0.10 filter-list route-map FilterLSAs
in</userinput>
switch(config-router)# <userinput>copy running-config startup-config</userinput>
```

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

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **router ospf***instance-tag* command to create a new OSPFv2 instance with the configured instance tag.

Example:

```
switch(config)# router ospf 201
switch(config-router)#
```

Step 3 Use the **area***area-id***stub** command to create this area as a stub area.

Example:

```
switch(config-router)# area 0.0.0.10
stub
```

Step 4 (Optional) Use the **area***area-id***default-cost***cost** command to set the cost metric for the default summary route sent into this stub area.

Example:

```
switch(config-router)# area 0.0.0.10 default-cost 25
```

The range is from 0 to 16777215. The default is 1.

Step 5 (Optional) Use the **show ip ospf***instance-tag* command to see OSPF information.

Example:

```
switch(config-router)# show ip ospf 201
```

Step 6 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

This example shows how to create a stub area:

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

Configure 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

Use the **area** *area-id* **stub no-summary** command to create this area as a totally stubby area.

Example:

```
switch(config-router)# area 20 stub
no-summary
```

Configure NSSA

You can configure an NSSA for part of an OSPFv2 domain where limited external traffic is required. 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.
- No summary—Blocks all summary routes from flooding the NSSA. Use this option on the NSSA ABR.
- 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.

Note The translate option requires a separate **area** *area-id* **nssa**command, preceded by the **area** *area-id* **nssa** command that creates the NSSA and configures the other options.

Note You can use command area 0.0.0.2 NSSA translate type7 to enable translate. Ensure that you configure command area 0.0.0.2 NSSA to designate Area 2 as NSSA.

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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

```
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* **nssa [no-redistribution] [default-information-originate]originate** [**route-map** *map-name*]] [**no-summary**]

Example:

```
switch(config-router)# area 0.0.0.10
nssa no-redistribution
```

Creates this area as an NSSA.

Step 4 (Optional) **area** *area-id* **nssa translate type7** {**always | never**} [**suppress-fa**]

Example:

```
switch(config-router)# area 0.0.0.10
nssa translate type7 always
```

Configures the NSSA to translate AS External (type 7) LSAs to NSSA External (type 5) LSAs.

Step 5 (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 6 (Optional) **show ip ospf** *instance-tag*

Example:

```
switch(config-router)# show ip ospf 201
```

Displays OSPF information.

Step 7 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

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

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

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

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router ospf 201</userinput>
switch(config-router)# <userinput>area 0.0.0.10 nssa default-info-originate</userinput>
switch(config-router)# <userinput>copy running-config startup-config</userinput>
```

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

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

This example shows how to create an NSSA and then configure the NSSA to always translate AS External (type 7) LSAs to NSSA External (type 5) LSAs:

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

Configure multi-area adjacency

You can add more than one area to an existing OSPFv2 interface. The additional logical interfaces support multi-area adjacency.

Before you begin

- You must enable OSPFv2 (see the Enabling OSPFv2 section).
- Ensure that you have configured a primary area for the interface (see the Configuring Networks in OSPFv2 section).

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface** *interface-type slot/port* command to enter interface configuration mode.

Example:

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

Step 3 Use the **ip router ospf** [*instance-tag*] **multi-area** *area-id* command to add the interface to another area.

Example:

```
switch(config-if)# ip router ospf 201 multi-area 3
```

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 that interface.

Step 4 (Optional) Use the **show ip ospf** *instance-tag* **interface** *interface-type slot/port* command to see OSPF information.

Example:

```
switch(config-if)# show ip ospf 201 interface ethernet 1/2
```

Step 5 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Example

This example shows how to add a second area to an OSPFv2 interface:

```
switch# configure terminal
                switch(config)# interface ethernet 1/2
                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)# ip router ospf 201 multi-area 20
                switch(config-if)# copy running-config startup-config
```

Configure virtual links

A virtual link connects an isolated area to the backbone area through an intermediate area. See Virtual Links for more information. 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 Enabling OSPFv2 for more information.

Procedure

Step 1 Use the **router ospf** *instance-tag* command to create a new OSPFv2 instance with the configured instance tag.

Example:

```
switch(config)# router ospf 201
switch(config-router)#
```

Step 2 Use the **area***area-id***virtual link***router-id* command to create one end of a virtual link to a remote router.

Example:

```
switch(config-router)# area 0.0.0.10
virtual-link 10.1.2.3
switch(config-router-vlink)#
```

You must create the virtual link on that remote router to complete the link.

Step 3 (Optional) Use the **show ip ospf virtual-link** [**brief**] command to see OSPF virtual link information.

Example:

```
switch(config-router-vlink)# show ip ospf
virtual-link
```

Step 4 (Optional) Us the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-router-vlink)# copy running-config
startup-config
```

Step 5 (Optional) Use this **authentication** [**key-chain** *key-id* **message-digest** | **null**] command to override area-based authentication for this virtual link.

Example:

```
switch(config-router-vlink)#
authentication message-digest
```

Step 6 (Optional) Use the **authentication-key** [**0** | **3**] *key* command to configure a simple password for this virtual link.

Example:

```
switch(config-router-vlink)#
authentication-key 0 mypass
```

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 7 (Optional) Use the **dead-interval** *seconds* command to configure the OSPFv2 dead interval, in seconds.

Example:

```
switch(config-router-vlink)#
dead-interval 50
```

The range is from 1 to 65535. The default is four times the hello interval, in seconds.

Step 8 (Optional) Use the **hello-interval** *seconds* command to configure the OSPFv2 hello interval, in seconds.

Example:

```
switch(config-router-vlink)#
hello-interval 25
```

The range is from 1 to 65535. The default is 10 seconds.

Step 9 (Optional) Use the **message-digest-key** *key-id* **md5** [**0** | **3**] *key* command to configure message digest authentication for this virtual link.

Example:

```
switch(config-router-vlink)#
message-digest-key 21 md5 0 mypass
```

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.

Step 10 (Optional) Use the **retransmit-interval** *seconds* command to configure the OSPFv2 retransmit interval, in seconds.

Example:

```
switch(config-router-vlink)#
retransmit-interval 50
```

The range is from 1 to 65535. The default is 5.

Step 11 (Optional) Use the **transmit-delay** *seconds* command to configure the OSPFv2 transmit-delay, in seconds.

Example:

```
switch(config-router-vlink)#
transmit-delay 2
```

The range is from 1 to 450. The default is 1.

Example

Note For OSPFv2, the key identifier in the key key-id command supports values from 0 to 255 only.

In Keychain only key 0-255 will be supported by OSPFv2.

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

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

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

Configure redistribution

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

For redistributing the default route, you must specify the following parameter:

- **default-information originate** - Creates a default route into this OSPF domain if the default route exists in the RIB.

Note Beginning with Cisco NX-OS Release 7.0(3)I7(6), if you redistribute default routes into OSPF, Cisco NX-OS requires the **default-information originate** command to successfully advertise the default route.

For non-default routes, you can configure the following optional parameters for route redistribution in OSPF:

- **default-metric** - Sets all redistributed routes to the same cost metric.

Before you begin

Enable the OSPF feature. See Enabling OSPFv2.

- Create the necessary route maps used for redistribution.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **router ospf** *instance-tag* command to create a new OSPFv2 instance with the configured instance tag.

Example:

```
switch(config)# router ospf 201
switch(config-router)#
```

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

Example:

```
switch(config-router)# redistribute bgp
route-map FilterExternalBGP
```



Note Beginning with Cisco NX-OS Release 7.0(3)I7(6), if you redistribute default routes into OSPF, Cisco NX-OS requires the **default-information originate** command to successfully advertise the default route.

Step 4 Use the **default-information originate** [**always**] [**route-map** *map-name*] command to create a default route into this OSPF domain if the default route exists in the RIB.

Example:

```
switch(config-router)#
default-information-originate route-map
DefaultRouteFilter
```

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 Use the **default-metric** [*cost*] command to set the cost metric for the redistributed routes.

Example:

```
switch(config-router)# default-metric 25
```

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) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

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
```

Limit 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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **router ospf** *instance-tag* command to create a new OSPFv2 instance with the configured instance tag.

Example:

```
switch(config)# router ospf 201
switch(config-router)#
```

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

Example:

```
switch(config-router)# redistribute bgp
route-map FilterExternalBGP
```

Step 4 Use the **redistribute maximum-prefix** *max* [*threshold*] [**warning-only | withdraw** [*num-retries timeout*]] command to specifie a maximum number of prefixes that OSPFv2 distributes.

Example:

```
switch(config-router)# redistribute
maximum-prefix 1000 75 warning-only
```

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) Use the **show running-config ospf** command to see the OSPFv2 configuration.

Example:

```
switch(config-router)# show
running-config ospf
```

Step 6 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

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

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

Configure 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

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

Procedure

Step 1 Use the **router ospf** *instance-tag* command to create a new OSPFv2 instance with the configured instance tag.

Example:

```
switch(config)# router ospf 201
switch(config-router)#
```

Step 2 Use the **area** *area-id* **range** *ip-prefix/length* [**no-advertise**] [**cost** *cost*] command to create a summary address on an ABR for a range of addresses and optionally do not advertise this summary address in a Network Summary (type 3) LSA.

Example:

```
switch(config-router)# area 0.0.0.10
range 10.3.0.0/16
```

The *cost* range is from 0 to 16777215.

Step 3 Use the **summary-address** *ip-prefix/length* [**no-advertise** | **tag** *tag*] command to create a summary address on an ASBR for a range of addresses and optionally assign a tag for this summary address that can be used for redistribution with route maps.

Example:

```
switch(config-router)# summary-address
10.5.0.0/16 tag 2
```

Step 4 (Optional) Use the **show ip ospf summary-address** command to see information about OSPF summary addresses.

Example:

```
switch(config-router)# show ip ospf
summary-address
```

Step 5 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-router)# copy running-config
startup-config
```

Example

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

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

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

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

Configure stub route advertisements

Use stub route advertisements when you want to limit the OSPFv2 traffic through this router for a short time. For more information, see the OSPFv2 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

Step 1 Use the **router ospf** *instance-tag* command to create a new OSPFv2 instance with the configured instance tag.

Example:

```
switch(config)# router ospf 201
switch(config-router)#
```

Step 2 Use the **max-metric router-lsa** [**external-lsa** [*max-metric-value*]] [**include-stub**] [**on-startup** {*seconds* | **wait-for bgp** *tag*}] [**summary-lsa** [*max-metric-value*}] command to configure OSPFv2 stub route advertisements.

Example:

```
switch(config-router)# max-metric
router-lsa
```

Step 3 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-router)# copy running-config
startup-config
```

Example

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

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

Configure the administrative distance of routes

You can set the administrative distance of routes added by OSPFv2 into the RIB.

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

- OSPF supports a table map to filter and change the distances of IPv4 and IPv6 prefixes.

Before you begin

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

- See the guidelines and limitations for this feature in the Guidelines and Limitations for OSPFv2 section.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **router ospf***instance-tag* command to create a new OSPFv2 instance with the configured instance tag.

Example:

```
switch(config)# router ospf 201
switch(config-router)#
```

Step 3 Use the [**no**] **table-map** *map-name* command to configure the policy for filtering or modifying OSPFv2 routes before sending them to the RIB.

Example:

```
switch(config-router)# table-map foo
```

You can enter up to 63 alphanumeric characters for the map name.

Step 4 Use the **exit** command to exit routes configuration mode.

Example:

```
switch(config-router)# exit
switch(config)#
```

Step 5 Use the **route-map** *map-name* [**permit** | **deny**] [*seq*] command to create a route map or enters route-map configuration mode for an existing route map.

Example:

```
switch(config)# route-map foo permit 10
switch(config-route-map)#
```

Use *seq* to order the entries in a route map.



Note The **permit** option enables you to set the distance. If you use the **deny** option, the default distance is applied.

Step 6 Use the **match route-type** *route-type* command to match the route types.

Example:

```
switch(config-route-map)# match
route-type external
```

These are the route types:

- external—The external route (BGP, EIGRP, and OSPF type 1 or 2)
- inter-area—OSPF inter-area route
- internal—The internal route (including the OSPF intra- or inter-area)
- intra-area—OSPF intra-area route
- nssa-external—The NSSA external route (OSPF type 1 or 2)
- type-1—The OSPF external type 1 route
- type-2—The OSPF external type 2 route

Step 7 Use the **match ip route-source prefix-list** *name* command to match the IPv4 route source address or router ID of a route to one or more IP prefix lists.

Example:

```
switch(config-route-map)# match
ip route-source prefix-list p1
```

Use the **ip prefix-list** command to create the prefix list.

Step 8 Use the **match ip address prefix-list** *name* command to match against one or more IPv4 prefix lists.

Example:

```
switch(config-route-map)# match
ip address prefix-list p1
```

Use the **ip prefix-list** command to create the prefix list.

Step 9 Use the **set distance** *value* command to set the administrative distance of routes for OSPFv2.

Example:

```
switch(config-route-map)# set distance
150
```

The range is from 1 to 255.

Step 10 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-route-map)# copy running-config
startup-config
```

Example

This example shows how to configure the OSPFv2 administrative distance for inter-area routes to 150, for external routes to 200, and for all prefixes in prefix list p1 to 190:

```
switch# <userinput>configure terminal</userinput>
 switch(config)# <userinput>router ospf 201</userinput>
 switch(config-router)# <userinput>table-map foo</userinput>
 switch(config-router)# <userinput>exit</userinput>
 switch(config)# <userinput>route-map foo permit 10</userinput>
 switch(config-route-map)# <userinput>match route-type inter-area</userinput>
 switch(config-route-map)# <userinput>set distance 150</userinput>
 switch(config-route-map)# <userinput>exit</userinput>
 switch(config)# <userinput>route-map foo permit 20</userinput>
 switch(config-route-map)# <userinput>match route-type external</userinput>
 switch(config-route-map)# <userinput>set distance 200</userinput>
 switch(config-route-map)# <userinput>exit</userinput>
 switch(config)# <userinput>route-map foo permit 30</userinput>
 switch(config-route-map)# <userinput>match ip route-source prefix-list p1</userinput>
 switch(config-route-map)# <userinput>match ip address prefix-list p1</userinput>
 switch(config-route-map)# <userinput>set distance 190</userinput>
```

Modify 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 the Flooding and LSA group pacing, on page 162 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

Step 1 Use the **router ospf** *instance-tag* command to

Example:

```
switch(config)# router ospf 201
 switch(config-router)#
```

Creates a new OSPFv2 instance with the configured instance tag.

Step 2 Use the **timers lsa-arrival** *msec* command to set the LSA arrival time in milliseconds.

Example:

```
switch(config-router)# timers
 lsa-arrival 2000
```

The range is from 10 to 600000. The default is 1000 milliseconds.

Step 3 Use the **timers lsa-group-pacing** *seconds* command to set the interval in seconds for grouping LSAs.

Example:

```
switch(config-router)# timers
 lsa-group-pacing 1800
```

The range is from 1 to 1800. The default is 240 seconds.

Step 4 Use the **timers throttle lsa** *start-time hold-interval max-time* command to set the rate limit in milliseconds for generating LSAs with the timers.

Example:

```
switch(config-router)# timers throttle
 lsa 3000 6000 6000
```

These are the timers:

- *start-time*—The range is from 0 to 5000 milliseconds. The default value is 0 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 5 Use the **timers throttle spf** *delay-time hold-time max-wait* command to set the SPF best path schedule in seconds between SPF best path calculations with timers.

Example:

```
switch(config-router)# timers throttle
 spf 3000 2000 4000
```

These are the timers:

- *delay-time*—The range is from 1 to 600,000 milliseconds. The default value is 200 milliseconds.

- *hold-time*—The range is from 1 to 600,000 milliseconds. The default value is 1000 milliseconds.
- *max-wait* —The range is from 1 to 600,000 milliseconds. The default value is 5000 milliseconds.

Step 6 Use the **interface** *type slot/port* command to enter interface configuration mode.

Example:

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

Step 7 Use the **ip ospf hello-interval** *seconds* command to set the hello interval for this interface.

Example:

```
switch(config-if)# ip ospf
 hello-interval 30
```

The range is from 1 to 65535. The default is 10.

Step 8 Use the **ip ospf dead-interval** *seconds* command to set the dead interval for this interface.

Example:

```
switch(config-if)# ip ospf dead-interval
 30
```

The range is from 1 to 65535.

Step 9 Use the **ip ospf retransmit-interval** *seconds* command to set the estimated time in seconds between LSAs transmitted from this interface.

Example:

```
switch(config-if)# ip ospf
 retransmit-interval 30
```

The range is from 1 to 65535. The default is 5.

Step 10 Use the **ip ospf transmit-delay** *seconds* command to set the estimated time in seconds to transmit an LSA to a neighbor.

Example:

```
switch(config-if)# ip ospf transmit-delay 450
 switch(config-if)#
```

The range is from 1 to 450. The default is 1.

Step 11 (Optional) Use the **show ip ospf** command to see information about OSPF.

Example:

```
switch(config-if)# show ip ospf
```

Step 12 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config
 startup-config
```

Example

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

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

Configure graceful restart

Graceful restart is enabled by default. You can configure the following optional parameters for graceful restart in an OSPFv2 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 OSPFv2 instance. OSPFv2 does not participate in the graceful restart of a neighbor.
- Planned graceful restart only—Configures OSPFv2 to support graceful restart only in the event of a planned restart.

Before you begin

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

- Ensure that all neighbors are configured for graceful restart with matching optional parameters set.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **router ospf***instance-tag* command to create a new OSPFv2 instance with the configured instance tag.

Example:

```
switch(config)# router ospf 201
 switch(config-router)#
```

Step 3 Use the **graceful-restart** command to enable a graceful restart.

Example:

```
switch(config-router)# graceful-restart
```

A graceful restart is enabled by default.

Step 4 (Optional) Use the **graceful-restart grace-period***seconds* command to set the grace period, in seconds.

Example:

```
switch(config-router)# graceful-restart
 grace-period 120
```

The range is from 5 to 1800. The default is 60 seconds.

Step 5 (Optional) Use the **graceful-restart helper-disable** command to disable helper mode.

Example:

```
switch(config-router)# graceful-restart
 helper-disable
```

This feature is enabled by default.

Step 6 (Optional) Use the **graceful-restart planned-only** command to configure a graceful restart for planned restarts only.

Example:

```
switch(config-router)# graceful-restart
 planned-only
```

Step 7 (Optional) Use the **show ip ospf***instance-tag* command to see OSPF information.

Example:

```
switch(config-router)# show ip ospf 201
```

Step 8 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

This example shows how to enable a graceful restart if it has been disabled and set the grace period to 120 seconds:

```
switch# <userinput>configure terminal</userinput>
 switch(config)# <userinput>router ospf 201</userinput>
 switch(config-router)# <userinput>graceful-restart</userinput>
 switch(config-router)# <userinput>graceful-restart grace-period 120</userinput>
 switch(config-router)# <userinput>copy running-config startup-config</userinput>
```

Restart 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

Use the **restart ospf** *instance-tag* command to restart the OSPFv2 instance and removes all neighbors.

Example:

```
switch(config)# restart ospf 201
```

Configure 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

Step 1 Use the **vrf context***vrf-name* command to create a new VRF and enters VRF configuration mode.

Example:

```
switch(config)# vrf context
 RemoteOfficeVRF
 switch(config-vrf)#
```

Step 2 Use the **router ospf***instance-tag* command to create a new OSPFv2 instance with the configured instance tag.

Example:

```
switch(config-vrf)# router ospf 201
 switch(config-router)#
```

Step 3 Use the **vrf***vrf-name* command to enter VRF configuration mode.

Example:

```
switch(config-router)# vrf
 RemoteOfficeVRF
 switch(config-router-vrf)#
```

Step 4 (Optional) Use the **maximum-paths***path*command to configure the maximum number of equal OSPFv2 paths to a destination in the route table for this VRF.

Example:

```
switch(config-router-vrf)# maximum-paths
 4
```

This feature is used for load balancing.

Step 5 Usee the **interface***interface-type slot/port* command to enter interface configuration mode.

Example:

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

Step 6 Use the **vrf member***vrf-name* command to add this interface to a VRF.

Example:

```
switch(config-if)# vrf member
 RemoteOfficeVRF
```

Step 7 Use the **ip address***ip-prefix/length* command to configure an IP address for this interface.

Example:

```
switch(config-if)# ip address
 192.0.2.1/16
```

You must do this step after you assign this interface to a VRF.

Step 8 Use the **ip router ospf***instance-tag***area***area-id* command to assign this interface to the OSPFv2 instance and area configured.

Example:

```
switch(config-if)# ip router ospf 201
 area 0
```

Step 9 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config
 startup-config
```

Example

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

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

Verify the OSPFv2 configuration

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

Command	Purpose
show ip ospf [*instance-tag*] [**vrf** *vrf-name*]	Displays information about one or more OSPF routing instances. The output includes the following area-level counts: • Interfaces in this area—A count of all interfaces added to this area (configured interfaces). • Active interfaces—A count of all interfaces considered to be in router link states and SPF (UP interfaces). • Passive interfaces—A count of all interfaces considered to be OSPF passive (no adjacencies will be formed). • Loopback interfaces—A count of all local loopback interfaces.
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 OSPFv2-related interface information.
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 the VRF-based OSPFv2 configuration.

Command	Purpose
show running-configuration ospf	Displays the current running OSPFv2 configuration.

Monitor OSPFv2

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 policy statistics redistribute {**bgp** *id* | **direct** | **eigrp** *id* | **isis** *id* | **ospf** *id* | **rip** *id* | **static**} [**vrf** {*vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv2 route policy statistics.
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
 ip router ospf 201 area 0.0.0.10
 ip ospf authentication
 ip ospf authentication-key 0 mypass
```

OSPF RFC compatibility mode example

The following example shows how to configure OSPF to be compatible with routers that comply with RFC 1583:

Note You must configure RFC 1583 compatibility on any VRF that connects to routers running only RFC 1583 compatible OSPF.

```
switch# configure terminal
 switch(config)# feature ospf
 switch(config)# router ospf Test1
 switch(config-router)# rfc1583compatibility
```

```
switch(config-router)# vrf A
switch(config-router-vrf)# rfc1583compatibility
```

Related Documents for OSPFv2

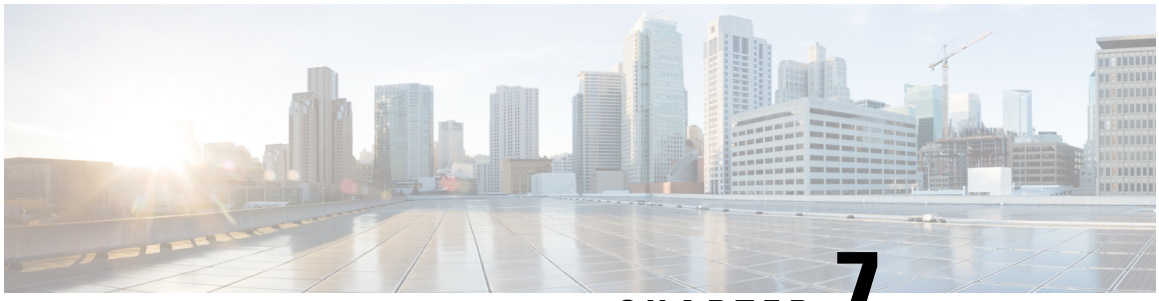
Related Topic	Document Title
Keychains	Cisco Nexus 9000 Series NX-OS Security Configuration Guide
OSPFv3 for IPv6 networks	OSPFv3, on page 157
Route maps	Configuring Route Policy Manager

Related Documents for OSPFv2

Related Topic	Document Title
Keychains	Cisco Nexus 9000 Series NX-OS Security Configuration Guide
OSPFv3 for IPv6 networks	OSPFv3, on page 157
Route maps	Configuring Route Policy Manager

MIBs

MIBs	MIBs Link
MIBs related to OSPFv2	To locate and download supported MIBs, go to the following URL: https://cisco.github.io/cisco-mibs/supportlists/nexus9000/Nexus9000MIBSupportList.html



CHAPTER 7

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 these sections:

- OSPFv3, on page 157
- Multi-Area adjacency, on page 163
- OSPFv3 and the IPv6 unicast RIB, on page 163
- Address family support, on page 164
- Authentication and encryption , on page 164
- Advanced features, on page 164
- Prerequisites for OSPFv3, on page 169
- Guidelines and limitations for OSPFv3, on page 169
- Default settings, on page 171
- Configure basic OSPFv3, on page 172
- Configure advanced OSPFv3, on page 177
- Configure Encryption and Authentication, on page 199
- Verify the OSPFv3 configuration, on page 210
- Monitor OSPFv3, on page 211
- Configuration examples for OSPFv3, on page 212
- Related topics, on page 212
- Additional references, on page 212

OSPFv3

OSPFv3 is a link-state routing protocol defined by the IETF, designed for IPv6 networks. Each OSPFv3 router periodically sends "hello" packets on its OSPF-enabled interfaces to discover other OSPFv3 routers. When a neighbor is detected, the routers compare information in the hello packets to verify configuration compatibility. If they are compatible, the routers form an adjacency, which means they synchronize their link-state databases to ensure identical OSPFv3 routing information. For more information, see Overview, on page 7.

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. Each router

then uses Dijkstra's Shortest Path First (SPF) algorithm to build its route table. To know more about convergence, see Convergence.

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 OSPFv2, on page 111.

Comparison of OSPFv3 and OSPFv2

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

These are the key differences between the OSPFv3 and OSPFv2 protocols:

- 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.
- OSPFv3 redefines LSA types.

Hello packets

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 these tasks:

- Neighbor discovery
- Keepalives
- Bidirectional communications
- Designated router election. For more information, see Designated Routers.

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 Neighbors for more information.

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

When an OSPFv3 interface has compatible configuration with a remote interface, both the interfaces are considered neighbors. The two OSPFv3 interfaces must match these criteria:

- Hello interval
- Dead interval
- Area ID. See the Areas.
- Optional capabilities

If there is a match, these 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.
- 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.
- 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 OSPFv3 neighbors form full adjacencies. The establishment of adjacency depends on the network type and the presence of a Designated Router (DR). In some cases, only selected neighbors become fully adjacent and exchange Link-State Advertisements (LSAs), while others do not. For more information, see the Designated Routers.

Adjacency is established using Database Description (DD) packets, Link State Request (LSR) packets, and Link State Update (LSU) packets in OSPFv3.

The DD packet includes the link-state advertisement (LSA) headers from the link-state database of the neighbor. See the Link-State Database for more information.

The local router compares these headers with its own link-state database and determines which LSAs are new or updated. The local router sends an LSR packet for each LSA that it needs new or updated information on. The neighbor responds with an LSU packet. This exchange continues until both routers have the same link-state information.

Designated routers

In OSPFv3 networks with multiple routers, managing link-state advertisements (LSAs) efficiently is essential. Without coordination, each router could flood the network with identical LSAs, leading to unnecessary traffic. To address this, OSPFv3 may designate a single router, called the Designated Router (DR), to coordinate LSA flooding and represent the network to the wider OSPFv3 area. A Backup Designated Router (BDR) is also elected to take over if the DR becomes unavailable. See the Areas for more information on OSPFv3 area.

Network types are as follows:

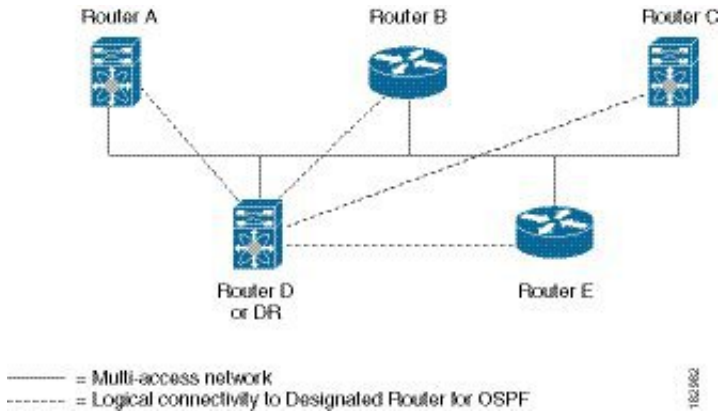
- Point-to-point:A network that exists only between two routers. All neighbors on a point-to-point network establish adjacency and there is no DR.
- Broadcast: A network with multiple routers that can communicate over a shared medium that allows broadcast traffic, such as Ethernet. OSPFv3 routers establish 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. Figure Figure 22: DR in Multi-Access Network , on page 160 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 22: DR in Multi-Access Network



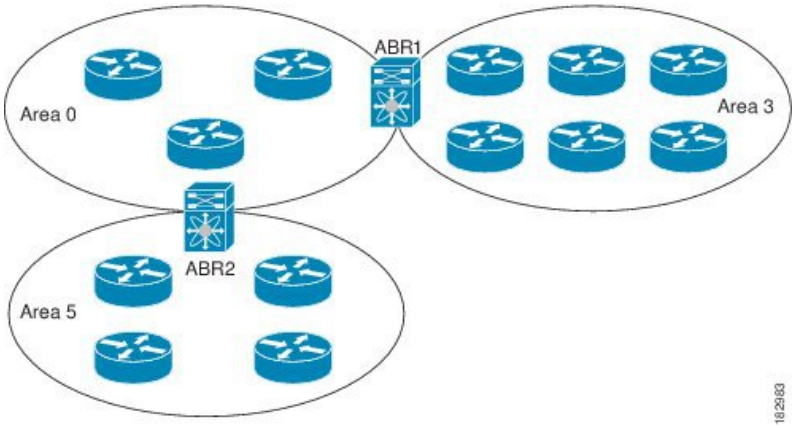
Areas

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. By dividing an OSPFv3 network into areas, you can limit the CPU and memory requirements that OSPFv3 puts on the routers.

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.

Figure 23: 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 from one connected area to the backbone area. See Route Summarization for more details.

The backbone area sends summarized information about one area to another area. In the figure Figure 23: OSPFv3 Areas, on page 161, Area 0 sends summarized information about Area 5 to Area 3.

OSPFv3 defines one other router type that is 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 Advanced Features.

Link-State advertisements

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

Link-State advertisement types

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

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

Type	Names	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 flooded to the local OSPFv3 area.
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.

Type	Names	Description
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.
11	Grace LSA	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 Gbps. 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 these 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 Areas to know more about area.

The LSAs are flooded based on the link-state refresh time. Each LSA has its own link-state refresh time. The default refresh time is 30 minutes.

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

This database contains all the collected link-state advertisements (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. Each router maintains a link-state database for the OSPFv3 network.

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.

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 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 Routing Information Base (RIB). OSPFv3 communicates with the IPv6 unicast RIB to do these:

- Add or remove routes.
- Handle route redistribution from other protocols.
- Provide convergence updates to remove stale OSPFv3 routes and for stub router advertisements. See Multiple OSPFv3 Instances for more information.

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 these:

- 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.

Authentication and encryption

You can configure authentication on OSPFv3 messages to prevent unauthorized or invalid routing updates in your network.

RFC 4552 provides authentication to OSPFv3 using an IPv6 Authentication Header (AH) or Encapsulating Security Payload (ESP) extension header. Beginning with Cisco NX-OS 7.0(3)I3(1), Cisco NX-OS supports RFC 4552 by using the IPv6 AH header to authenticate OSPFv3 packets.

Cisco NX-OS supports the IP Security (IPSec) authentication method and the Message Digest 5 (MD5) or Secure Hash Algorithm 1 (SHA-1) algorithm to authenticate OSPFv3 packets. OSPFv3 IPSec authentication supports static keys using commands.

Cisco NX-OS also supports the IPSec ESP method for both encryption and authentication of OSPFv3 messages. Encryption supports AES or 3DES algorithm for ESP encryption and SHA-1 or NULL for ESP authentication.

Beginning with Cisco NX-OS Release 10.4(1)F, Cisco NX-OS supports configuring encryption or authentication algorithms and keys using the keychain option.

You can configure IPSec encryption or authentication for an OSPFv3 process, an area, and/or an interface. The authentication configuration is inherited from process to area to interface level. If authentication is configured at all three levels, the interface configuration takes precedence over an area configurations, and an area configuration takes precedence over the process level.

Advanced features

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

Stub areas

A stub area is an area that does not allow AS External (type 5) LSAs. These LSAs are usually flooded throughout the local autonomous system to propagate external route information. You can limit the amount

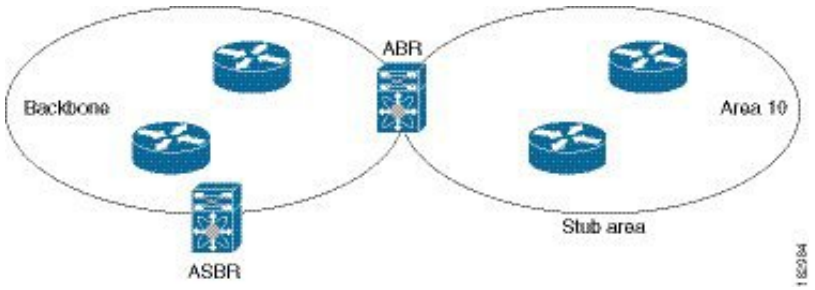
of external routing information that floods an area by making it a stub area. See Link-State advertisements, on page 161 to know more about LSA.

These are the requirements for stub area:

- 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.

The figure Figure 20: Stub area, on page 114 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 24: 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 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 ABR then floods these AS External LSAs throughout the OSPFv2 autonomous system. Summarization and filtering are supported during the translation. See Link-State advertisements, on page 161 for information about NSSA External 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 section).

The backbone Area 0 cannot be an NSSA

Note Beginning with Cisco NX-OS Release 9.3(1), OSPF became compliant with RFC 3101 section 2.5(3). When an Area Border Router attached to a Not-so-Stubby Area receives a default route LSA with P-bit clear, it should be ignored. OSPF had been previously adding the default route under these conditions.

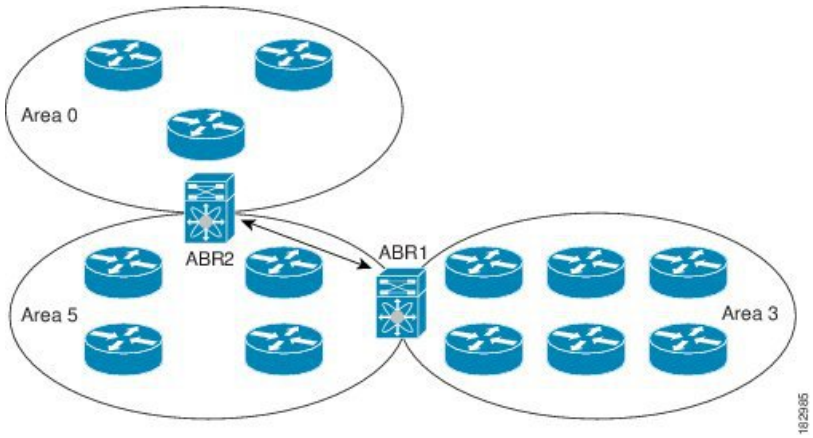
If you have already designed your networks with RFC non-compliant behavior and expect a default route to be added on NSSA ABR, you will see a change in behavior when you upgrade to Cisco NX-OS Release 9.3(1) and later.

If you decide to continue with the old behavior, you have the option to enable it with the **default-route nssa-abr pbit-clear** command. This command was implemented in Cisco NX-OS Release 9.3(1).

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. The figure Figure 25: Virtual Links, on page 166 shows a virtual link that connects Area 3 to the backbone area through Area 5.

Figure 25: 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

Route redistribution is a processes that allows OSPFv3 to learn routes from other routing protocols. You can configure OSPFv3 to assign a specific link cost to these redistributed routes or apply a default link cost to all of them.

OSPFv3 can learn routes from other routing protocols by using route redistribution. See the Route Redistribution, on page 12. 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 OSPFv3. 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, on page 523.

Route summarization

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.

You can use route summarization to reduce the number of unique routes that are flooded to every OSPF-enabled router because OSPFv3 shares all learned routes with every OSPF-enabled router.

Area border router (ABR) summarization is the process of aggregating routing information at the boundaries between OSPF areas. You typically perform summarization at ABRs to reduce the size of routing tables and improve network efficiency. Although you can configure summarization between any two areas, it is best to summarize routes toward the backbone area. This approach ensures that the backbone receives all summarized addresses and then injects these aggregates into other areas, optimizing routing updates and minimizing unnecessary detail across the network. There are two types of summarization:

- Inter-area route summarization
- External route summarization

Inter-area 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: It is specific to external routes that are injected into OSPFv3 using route redistribution. 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

High availabilities in Cisco NX-OS provide a multilevel architecture that ensures continuous network operation and minimal downtime. One key protocol supporting this is OSPFv3, which includes mechanisms for stateful restart and graceful restart to maintain routing stability during process interruptions.

OSPFv3 supports stateful restart, which is also referred to as non-stop routing (NSR). If OSPFv3 experiences problems, it attempts to restart from its previous run-time state. The neighbors do not register any neighbor event in this case. If the first restart is not successful and another problem occurs, OSPFv3 attempts a graceful restart.

A graceful restart, or non-stop forwarding (NSF), allows OSPFv3 to remain in the data forwarding path through a process restart. When OSPFv3 needs to perform a graceful restart, it 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 was 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 the graceful restart has finished.

Stateful restart is used in these scenarios:

- First recovery attempt after the process experiences problems
- User-initiated switchover using the **system switchover** command

Graceful restart is used in these scenarios:

- Second recovery attempt after the process experiences problems within a 4-minute interval
- Manual restart of the process using the **restart ospfv3** command
- Active supervisor removal
- Active supervisor reload using the **reload module** *active-sup* command

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. For the number of supported OSPFv3 instances, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

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

The shortest path first (SPF) algorithm optimization in Cisco NX-OS is a set of techniques that

- improve the efficiency of SPF calculations,
- reduce CPU load during routing updates, and
- speed up network convergence.

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

Bidirectional forwarding detection (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. See the Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide for more information.

Virtualization support

Cisco NX-OS supports multiple process instances of OSPFv3. Each OSPFv3 instance can support multiple virtual routing and forwarding (VRF) instances, up to the system limit. For the number of supported OSPFv3 instances, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

Prerequisites for OSPFv3

These are the prerequisites for OSPFv3:

- 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 topic.
- You are familiar with IPv6 addressing and basic configuration. See IPv6 Addresses, on page 57 for information on IPv6 routing and addressing.

Guidelines and limitations for OSPFv3

OSPFv3 has the following configuration guidelines and limitations:

- The **graceful-restart planned-only** command under OSPFv2 on reload converts to the **graceful-restart** command.

  This is not causing any impact on the functionality. If the **graceful-restart planned-only** is not in the configuration, this problem is not applicable for that device.

  This occurs when the Cisco NX-OS release is 9.3(2) and CSCvs57583 is not included in the release. A workaround is to unconfigure the **graceful-restart** command and reconfigure the old command.

- Names in the prefix-list are case-insensitive. We recommend using unique names. Do not use the same name by modifying uppercase and lowercase characters. For example, CTCPrimaryNetworks and CtcPrimaryNetworks are not two different entries.

- If you enter the **no graceful-restart planned only** command, graceful restart is disabled.
- 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
```

- In scaled scenarios, when the number of interfaces and link-state advertisements in an OSPF process is large, the snmp-walk on OSPF MIB objects is expected to time out with a small-values timeout at the SNMP agent. If you observe a timeout on the querying SNMP agent while polling OSPF MIB objects, increase the timeout value on the polling SNMP agent.
- The following guidelines and limitations apply to the administrative distance feature:
  - When an OSPF route has two or more equal cost paths, configuring the administrative distance is non-deterministic for the **match ip route-source** command.
  - For matching route sources in OSPFv3 routes, you must configure **match ip route-source** instead of **match ipv6 route-source** because the route sources and router IDs for OSPFv3 are IPv4 addresses.
  - Configuring the administrative distance is supported only for the **match route-type**, **match ipv6 address prefix-list**, and **match ip route-source prefix-list** commands. The other match statements are ignored.
  - The discard route is always assigned an administrative distance of 220. No configuration in the table map applies to OSPF discard routes.
  - There is no preference among the **match route-type**, **match ipv6 address**, and **match ip route-source** commands for setting the administrative distance of OSPF routes. In this way, the behavior of the table map for setting the administrative distance in Cisco NX-OS OSPF is different from the behavior in Cisco IOS OSPF.
- If you configure the **delay restore** *seconds* command in vPC configuration mode and if the VLANs on the multichassis EtherChannel trunk (MCT) are announced by OSPFv2 or OSPFv3 using switch virtual interfaces (SVIs), those SVIs are announced with MAX_LINK_COST on the vPC secondary node during the configured time. As a result, all route or host programming completes after the vPC synchronization operation (on a peer reload of the secondary vPC node) before attracting traffic. This behavior allows for minimal packet loss for any north-to-south traffic.
- If you configure the same *area-id* for the primary area and any multiarea, the configuration is accepted without displaying an error. When you configure the primary area and any multiareas, do not use the same *area-id.*



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.

- If you use the **network** *ip address mask* command under OSPF, an error message will be displayed, and you will be prompted to enable OSPF under an interface with **area** *area id* command.

- It is recommended that you use the OSPF default timers (hello-interval:10 and dead-interval:40). For better convergence time, you can use the BFD along with OSPF. This combination will give sub-second link/adjacency flaps detection and very low convergence time.
- While OSPF support are aggressive timers, these are not commended as aggressive timers will bring the adjancency down quickly as well as cause CPU churn. We recommend you to use the default timers and use BFD (Bidirectional Forwarding Detection) to get sub-second failure detection.
- The Route Policy Manager (RPM) does not support IPv6 redistribution for filtering BGP route updates based on the BGP community attribute using community lists in a route map. This functionality is only available for IPv4 redistribution.
- Beginning with Cisco NX-OS Release 10.3(1)F, OSPFv3 is supported on the Cisco Nexus 9808 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, OSPFv3 is supported on the Cisco Nexus 9804 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, the keychain support is provided for OSPFv3 encryption and authentication commands on the Cisco NX-OS switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, OSPFv3 is supported on Cisco Nexus X98900CD-A and X9836DM-A line cards with 9808 and 9804 switches.
- Beginning with Cisco NX-OS Release 10.4(2)F, OSPFv3 is supported on Cisco Nexus 9232E-B1 switches.
- Beginning with Cisco NX-OS Release 10.6(1)F, OSPFv3 is supported on Cisco Nexus N9336C-SE1 switches.
- Beginning with Cisco NX-OS Release 10.6(2)F, BFD (Single Hop) and OSPFv3 are supported on Cisco Nexus N9324C-SE1U and N9348Y2C6D-SE1U platform switches.

Default settings

The table lists the default settings for OSPFv3 parameters.

Table 19: Default OSPFv3 Parameters

Parameters	Default
Administrative distance	110
Hello interval	10 seconds
Dead interval	40 seconds
Discard routes	Enabled
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

Parameters	Default
LSA minimal arrival time	1000 milliseconds
LSA group pacing	10 seconds
SPF calculation initial delay time	200 milliseconds
SPF calculation minimum hold time	1000 milliseconds
SPF calculation maximum wait time	5000 milliseconds

Configure basic OSPFv3

Configure OSPFv3 after you have designed your OSPFv3 network.

Enable OSPFv3

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **feature ospfv3** command to enable OSPFv3.

Example:

```
switch(config)# feature ospfv3
```

Using the **no** keyword with this command disables the OSPFv3 feature and removes all associated configuration.

Step 3 (Optional) Use the **show feature** command to see see enabled and disabled features.

Example:

```
switch(config)# show feature
```

Step 4 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config
startup-config
```

Create 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. , 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.
- Name lookup—Translates OSPF router IDs to hostnames, either by looking up the local hosts database or querying DNS names in IPv6.
- 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 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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the [**no**] **router ospfv3** *instance-tag* command to create a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 201
 switch(config-router)#
```

Note The **no router ospfv3** *instance tag* command does not remove OSPF configuration in interface mode. You must manually remove any OSPFv3 commands configured in interface mode.

Step 3 (Optional) Use the **router-id** *ip-address* command to configures the OSPFv3 router ID.

Example:

```
switch(config-router)# router-id
 192.0.2.1
```

This ID uses the dotted decimal notation and identifies this OSPFv3 instance and must exist on a configured interface in the system.

Step 4 (Optional) Use the **show ipv6 ospfv3** *instance-tag* command to see OSPFv3 information.

Example:

```
switch(config-router)# show ipv6 ospfv3
 201
```

Step 5 (Optional) Use the **log-adjacency-changes** [**detail**] command to generate a system message whenever a neighbor changes state.

Example:

```
switch(config-router)#
 log-adjacency-changes
```

Step 6 (Optional) Use the **passive-interface default** command to suppress routing updates on all interfaces.

Example:

```
switch(config-router)# passive-interface
 default
```

This command is overridden by the VRF or interface command mode configuration.

Step 7 (Optional) Use the **distance** *number* command to configure the administrative distance for this OSPFv3 instance.

Example:

```
switch(config-router-af)# distance 25
```

The range is from 1 to 255. The default is 110.

Step 8 (Optional) Use the **maximum-paths** *paths* command to configure the maximum number of equal OSPFv3 paths to a destination in the route table.

Example:

```
switch(config-router-af)# maximum-paths
 4
```

The range is from 1 to 16. The default is 8. This command is used for load balancing.

Step 9 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

This example shows how to create an OSPFv3 instance:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router ospfv3 201</userinput>
 switch(config-router)# <userinput>copy running-config startup-config</userinput>
```

Configure 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.

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

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 section).

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface***interface-type slot/port* command to enter interface configuration mode.

Example:

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

Step 3 Use the **ipv6 address***ipv6-prefix/length* command to assign an IPv6 address to this interface.

Example:

```
switch(config-if)# ipv6 address
 2001:0DB8::1/48
```

Step 4 Use the **ipv6 router ospfv3***instance-tag***area***area-id* [**secondaries none**] command to add the interface to the OSPFv3 instance and area.

Example:

```
switch(config-if)# ipv6 router ospfv3
 201 area 0
```

Step 5 (Optional) Use the **show ipv6 ospfv3***instance-tag***interface***interface-type slot/port* command to see OSPFv3 information.

Example:

```
switch(config-if)# show ipv6 ospfv3 201
 interface ethernet 1/2
```

Step 6 (Optional) Use the **ospfv3 cost***number* command to configure the OSPFv3 cost metric for this interface.

Example:

```
switch(config-if)# ospfv3 cost 25
```

The default is to calculate a cost metric, based on the reference bandwidth and interface bandwidth. The range is from 1 to 65535.

Step 7 (Optional) Use the **ospfv3 dead-interval***seconds* command to configure the OSPFv3 dead interval, in seconds.

Example:

```
switch(config-if)# ospfv3 dead-interval 50
```

The range is from 1 to 65535. The default is four times the hello interval, in seconds.

Step 8 (Optional) Use the **ospfv3 hello-interval***seconds* command to configure the OSPFv3 hello interval, in seconds.

Example:

```
switch(config-if)# ospfv3 hello-interval
 25
```

The range is from 1 to 65535. The default is 10 seconds.

Step 9 (Optional) Use the **ospfv3 instance***instance* command to configure the OSPFv3 instance ID.

Example:

```
switch(config-if)# ospfv3 instance 25
```

The range is from 0 to 255. The default is 0. The instance ID is link-local in scope.

Step 10 (Optional) Use the **ospfv3 mtu-ignore** command to configure OSPFv3 to ignore any IP maximum transmission unit (MTU) mismatch with a neighbor.

Example:

```
switch(config-if)# ospfv3 mtu-ignore
```

The default is to not establish adjacency if the neighbor MTU does not match the local interface MTU.

Step 11 (Optional) Use the **ospfv3 network** {**broadcast** | **point-point**} command to set the OSPFv3 network type.

Example:

```
switch(config-if)# ospfv3 network
 broadcast
```

Step 12 (Optional) Use the [**default** | **no**] **ospfv3 passive-interface** command to suppress routing updates on the interface.

Example:

```
switch(config-if)# ospfv3
 passive-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.

Step 13 (Optional) Use the **ospfv3 priority***number* command to configure the OSPFv3 priority, used to determine the DR for an area.

Example:

```
switch(config-if)# ospfv3 priority 25
```

The range is from 0 to 255. The default is 1. See the Designated Routers section.

Step 14 (Optional) Use the **ospfv3 shutdown** command to shut down the OSPFv3 instance on this interface.

Example:

```
switch(config-if)# ospfv3 shutdown
```

Step 15 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

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 router ospfv3 201 area 0.0.0.10
 switch(config-if)# copy running-config startup-config
```

Configure advanced OSPFv3

Configure OSPFv3 after you have designed your OSPFv3 network.

Configure 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, on page 523.

Procedure

Step 1 Use the **router ospfv3** *instance-tag* command to create a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 201
 switch(config-router)#
```

Step 2 Use the **address-family ipv6 unicast** command to enter IPv6 unicast address family mode.

Example:

```
switch(config-router)# address-family
 ipv6 unicast
 switch(config-router-af)#
```

Step 3 Use the **area** *area-id* **filter-list route-map** *map-name* {**in | out**} command to filter incoming or outgoing Inter-Area Prefix (type 3) LSAs on an ABR.

Example:

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

Step 4 (Optional) Use the **show ipv6 ospfv3 policy statistics area** *id* **filter-list** {**in** | **out**} command to see OSPFv3 policy information.

Example:

```
switch(config-router-af)# show ipv6 ospfv3
 policy statistics area 0.0.0.10
 filter-list in
```

Step 5 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-router-af)# copy running-config
 startup-config
```

Example

This example shows how to configure a filter list for a route map:

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

Configure 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 section).
- Ensure that there are no virtual links or ASBRs in the proposed stub area.

Procedure

Step 1 Use the **router ospfv3** *instance-tag* command to create a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 201
 switch(config-router)#
```

Step 2 Use the **area** *area-id* **stub** command to create this area as a stub area.

Example:

```
switch(config-router)# area 0.0.0.10
 stub
```

Step 3 (Optional) Use the **address-family ipv6 unicast** command to enter IPv6 unicast address family mode.

Example:

```
switch(config-router)# address-family
 ipv6 unicast
 switch(config-router-af)#
```

Step 4 (Optional) Use the **area** *area-id* **default cost** *cost* command to set the cost metric for the default summary route sent into this stub area.

Example:

```
switch(config-router-af)# area 0.0.0.10
 default-cost 25
```

The range is from 0 to 16777215.

Step 5 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-router-af)# copy running-config
 startup-config
```

Example

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

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

Configure 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

Use the **area** *area-id* **stub no-summary** command to create this area as a totally stubby area.

Example:

```
switch(config-router)# area 20 stub
 no-summary
```

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

Note The translate option requires a separate **area***area-id***nssa** command, preceded by the **area***area-id***nssa** command that creates the NSSA and configures the other options.

Before you begin

You must enable OSPF (see the Enabling OSPFv3 section).

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

Procedure

Step 1 Use the **router ospfv3***instance-tag* command to create a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 201
  switch(config-router)#
```

Step 2 Use the **area***area-id***nssa** [**no-redistribution**][**default-information-originate**] [**route-map***map-name*] [**no-summary**] command to create this area as an NSSA.

Example:

```
switch(config-router)# area 0.0.0.10
  nssa
```

Step 3 (Optional) Use the **area***area-id***nssa translate type7** {**always | never**} [**suppress-fa**] command to configure the NSSA to translate AS External (type 7) LSAs to NSSA External (type 5) LSAs.

Example:

```
switch(config-router)# area 0.0.0.10
  nssa translate type7 always
```

Step 4 (Optional) Use the **address-family ipv6 unicast** command to enter IPv6 unicast address family mode.

Example:

```
switch(config-router)# address-family
  ipv6 unicast
  switch(config-router-af)#
```

Step 5 (Optional) Use the **area***area-id***default cost***cost* command to set the cost metric for the default summary route sent into this NSSA.

Example:

```
switch(config-router-af)# area 0.0.0.10
  default-cost 25
```

The range is from 0 to 16777215.

Step 6 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-router-af)# copy running-config
  startup-config
```

Example

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

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

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

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

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

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

<!--CSCvv84492-->This example shows how to create an NSSA and then configure the NSSA to always translate AS External (type 7) LSAs to NSSA External (type 5) LSAs:

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

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

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

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

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface** *interface-type slot/port* command to enter interface configuration mode.

Example:

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

Step 3 Use the **ipv6 router ospfv3** *instance-tag* **multi-area** *area-id* command to add the interface to another area.

Example:

```
switch(config-if)# ipv6 router ospfv3
  201 multi-area 3
```

Step 4 (Optional) Use the **show ipv6 ospfv3** *instance-tag* **interface** *interface-type slot/port* command to see OSPFv3 information.

Example:

```
switch(config-if)# show ipv6 ospfv3 201
  interface ethernet 1/2
```

Step 5 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

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

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

Configure 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. For more information, see the Enabling OSPFv3.

Procedure

Step 1 Use the **router ospfv3** *instance-tag* command to create a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 201
  switch(config-router)#
```

Step 2 Use the **area** *area-id* **virtual-link** *router-id* command to create one end of a virtual link to a remote router.

Example:

```
switch(config-router)# area 0.0.0.10
  virtual-link 2001:0DB8::1
  switch(config-router-vlink)#
```

You must create the virtual link on that remote router to complete the link.

Step 3 (Optional) Use the **show ipv6 ospfv3 virtual-link** [**brief**] command to see see OSPFv3 virtual link information.

Example:

```
switch(config-router-vlink)# show ipv6 ospfv3
  virtual-link
```

Step 4 (Optional) Use the **dead-interval** *seconds* command to configure the OSPFv3 dead interval, in seconds.

Example:

```
switch(config-router-vlink)#
  dead-interval 50
```

The range is from 1 to 65535. The default is four times the hello interval, in seconds.

Step 5 (Optional) Use the **hello-interval** *seconds* command to configure the OSPFv3 hello interval, in seconds.

Example:

```
switch(config-router-vlink)#
  hello-interval 25
```

The range is from 1 to 65535. The default is 10 seconds.

Step 6 (Optional) Use the **retransmit-interval** *seconds* command to configure the OSPFv3 retransmit interval, in seconds.

Example:

```
switch(config-router-vlink)#
  retransmit-interval 50
```

The range is from 1 to 65535. The default is 5.

Step 7 (Optional) Use the **transmit-delay** *seconds* command to configure the OSPFv3 transmit-delay, in seconds.

Example:

```
switch(config-router-vlink)#
  transmit-delay 2
```

The range is from 1 to 450. The default is 1.

Step 8 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-router-vlink)#
  copy running-config startup-config
```

Example

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# <userinput>configure terminal</userinput>
 switch(config)# <userinput>router ospfv3 201</userinput>
 switch(config-router)# <userinput>area 0.0.0.10 virtual-link 2001:0DB8::10</userinput>
 switch(config-router-vlink)# <userinput>copy running-config startup-config</userinput>
```

Configuration for ABR 2 (router ID 2001:0DB8::10) is as follows:

```
switch# <userinput>configure terminal</userinput>
 switch(config)# <userinput>router ospfv3 201</userinput>
 switch(config-router)# <userinput>area 0.0.0.10 virtual-link 2001:0DB8::1</userinput>
 switch(config-router-vlink)# <userinput>copy running-config startup-config</userinput>
```

Configure 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 starting in 7.0(3)I7(6).

Before you begin

You must enable OSPF (see the Enabling OSPFv3 section).

- Create the necessary route maps used for redistribution.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **router ospfv3***instance-tag* command to create a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 201
  switch(config-router)#
```

Step 3 Use the **address-family ipv6 unicast** command to enter IPv6 unicast address family mode.

Example:

```
switch(config-router)# address-family
  ipv6 unicast
  switch(config-router-af)#
```

Step 4 Use the **redistribute** {**bgp***id* **| direct | isis** *id* **| rip** *id* **| static | dhcpv6**} **route-map** *map-name* command to redistribute the selected protocol into OSPFv3 through the configured route map.

Example:

```
switch(config-router-af)# redistribute
  bgp route-map FilterExternalBGP
```



Note If you redistribute static routes, Cisco NX-OS requires the **default-information originate** command to successfully redistribute the default static route starting in 7.0(3)I7(6).

Step 5 Use the **default-information originate** [**always**] [**route-map** *map-name*] command to create a default route into this OSPFv3 domain if the default route exists in the RIB.

Example:

```
switch(config-router-af)#
  default-information-originate route-map DefaultRouteFilter
```

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.



Note This command ignores **match** statements in the route map.

Step 6 Use the **default-metric** *cost* command to set the cost metric for the redistributed routes.

Example:

```
switch(config-router-af)# default-metric
  25
```

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 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-router-af)# copy running-config
  startup-config
```

Example

This example shows how to redistribute the Border Gateway Protocol (BGP) into OSPFv3:

```
switch# <userinput>configure terminal</userinput>
 switch(config)# <userinput>router ospfv3 201</userinput>
 switch(config-router)# <userinput>address-family ipv6 unicast</userinput>
 switch(config-router-af)# <userinput>redistribute bgp route-map FilterExternalBGP</userinput>

 switch(config-router-af)# <userinput>copy running-config startup-config</userinput>
```

Limit 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 section).

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **router ospfv3** *instance-tag* command to create a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 201
  switch(config-router)#
```

Step 3 Use the **address-family ipv6 unicast** command to enter IPv6 unicast address family mode.

Example:

```
switch(config-router)# address-family
  ipv6 unicast
  switch(config-router-af)#
```

Step 4 Use the **redistribute** {**bgp***id* | **direct** | **isis** *id* | **rip** *id* | **static**} **route-map** *map-name* command to redistribute the selected protocol into OSPFv3 through the configured route map.

Example:

```
switch(config-router-af)# redistribute
  bgp route-map FilterExternalBGP
```

Step 5 Use the **redistribute maximum-prefix***max* [*threshold*] [**warning-only** | **withdraw** [*num-retries timemout*]] command to specify a maximum number of prefixes that OSPFv2 distributes.

Example:

```
switch(config-router-af)# redistribute
  maximum-prefix 1000 75 warning-only
```

The range is from 0 to 65536. Optionally, specifies the following:

- *threshold*—Percent 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 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) Use the **show running-config ospfv3** command to see the OSPFv3 configuration.

Example:

```
switch(config-router-af)# show
 running-config ospf
```

Step 7 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

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

```
switch# <userinput>configure terminal</userinput>
 switch(config)# <userinput>router ospfv3 201</userinput>
 switch(config-router)# <userinput>address-family ipv6 unicast</userinput>
 switch(config-router-af)# <userinput>redistribute bgp route-map FilterExternalBGP</userinput>

 switch(config-router-af)# <userinput>redistribute maximum-prefix 1000 75</userinput>
```

Configure route summarization

You can configure route summarization for inter-area networks 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 section).

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **router ospfv3***instance-tag* command to Creates a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 201
switch(config-router)#
```

Step 3 Use the **address-family ipv6 unicast** command to enter IPv6 unicast address family mode.

Example:

```
switch(config-router)# address-family
ipv6 unicast
switch(config-router-af)#
```

Step 4 Use the **area** *area-id* **range** *ipv6-prefix/length* [**no-advertise**] [**cost** *cost*] command to create a summary address on an ABR for a range of addresses and optionally advertises this summary address in a Inter-Area Prefix (type 3) LSA.

Example:

```
switch(config-router-af)# area 0.0.0.10
range 2001:0DB8::/48 advertise
```

The cost range is from 0 to 16777215.

Step 5 Use the **summary-address** *ipv6-prefix/length* [**no-advertise**] [**tag** *tag*] command to

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) Use the **show ipv6 ospfv3 summary-address** command to see the information about OSPFv3 summary addresses.

Example:

```
switch(config-router-af)# show ipv6 ospfv3
summary-address
```

Step 7 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

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

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router ospfv3 201</userinput>
switch(config-router)# <userinput>address-family ipv6 unicast</userinput>
switch(config-router-af)# <userinput>area 0.0.0.10 range 2001:0DB8::/48</userinput>
switch(config-router-af)# <userinput>copy running-config startup-config</userinput>
```

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

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router ospfv3 201</userinput>
switch(config-router)# <userinput>address-family ipv6 unicast</userinput>
switch(config-router-af)# <userinput>summary-address 2001:0DB8::/48</userinput>
switch(config-router-af)# <userinput>no discard route internal</userinput>
switch(config-router-af)# <userinput>copy running-config startup-config</userinput>
```

Configure the administrative distance of routes

You can set the administrative distance of routes added by OSPFv3 into the RIB.

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

Before you begin

Ensure that you have enabled OSPF (see the OSPFv3, on page 157 section).

- See the guidelines and limitations for this feature in the Guidelines and limitations for OSPFv3, on page 169 section.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **router ospfv3** *instance-tag* command to create a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 201
switch(config-router)#
```

Step 3 Use the **address-family ipv6 unicast** command to enter IPv6 unicast address family mode.

Example:

```
switch(config-router)# address-family
ipv6 unicast
switch(config-router-af)#
```

Step 4 Use the [**no**] **table-map** *map-name* command to configure the policy for filtering or modifying OSPFv3 routes before sending them to the RIB.

Example:

```
switch(config-router-af)# table-map foo
```

You can enter up to 63 alphanumeric characters for the map name.

Step 5 Use the **exit** command to exit router address-family configuration mode.

Example:

```
switch(config-router-af)# exit
switch(config-router)#
```

Step 6 Use the **exit** command to exit router configuration mode.

Example:

```
switch(config-router)# exit
switch(config)#
```

Step 7 Use the **route-map** *map-name* [**permit** | **deny**] [*seq*] command to create a route map or enters route-map configuration mode for an existing route map.

Example:

```
switch(config)# route-map foo permit 10
switch(config-route-map)#
```

Use *seq* to order the entries in a route map.

Note The **permit** option enables you to set the distance. If you use the **deny** option, the default distance is applied.

Step 8 Use the **match route-type** *route-type* command to match against route type.

Example:

```
switch(config-route-map)# match
route-type external
```

These are the route types:

- external—The external route (BGP, EIGRP, and OSPF type 1 or 2)
- inter-area—The OSPF inter-area route
- internal—The internal route (including the OSPF intra- or inter-area)
- intra-area—The OSPF intra-area route
- nssa-external—The NSSA external route (OSPF type 1 or 2)
- type-1—The OSPF external type 1 route
- type-2—The OSPF external type 2 route

Step 9 Use the **match ip route-source prefix-list** *name* command to match the IPv6 route source address or router ID of a route to one or more IP prefix lists.

Example:

```
switch(config-route-map)# match ip
route-source prefix-list p1
```

Use the **ip prefix-list** command to create the prefix list.



Note For OSPFv3, the router ID is 4 bytes.

Step 10 Use the **match ipv6 address prefix-list** *name* command to match against one or more IPv6 prefix lists.

Example:

```
switch(config-route-map)# match ipv6
address prefix-list p1
```

Use the **ip prefix-list** command to create the prefix list.

Step 11 Use the **set distance** *value* command to set the administrative distance of routes for OSPFv3.

Example:

```
switch(config-route-map)# set distance
150
```

The range is from 1 to 255.

Step 12 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-route-map)# copy
running-config startup-config
```

Example

This example shows how to configure the OSPFv3 administrative distance for inter-area routes to 150, for external routes to 200, and for all prefixes in prefix list p1 to 190:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router ospfv3 201</userinput>
```

```
switch(config-router)# <userinput>address-family ipv6 unicast</userinput>
switch(config-router-af)# <userinput>table-map foo</userinput>
switch(config-router)# <userinput>exit</userinput>
switch(config)# <userinput>exit</userinput>
switch(config)# <userinput>route-map foo permit 10</userinput>
switch(config-route-map)# <userinput>match route-type inter-area</userinput>
switch(config-route-map)# <userinput>set distance 150</userinput>
switch(config)# <userinput>route-map foo permit 20</userinput>
switch(config-route-map)# <userinput>match route-type external</userinput>
switch(config-route-map)# <userinput>set distance 200</userinput>
switch(config)# <userinput>route-map foo permit 30</userinput>
switch(config-route-map)# <userinput>match ip route-source prefix-list p1</userinput>
switch(config-route-map)# <userinput>match ipv6 address prefix-list p1</userinput>
switch(config-route-map)# <userinput>set distance 190</userinput>
switch(config-route-map)# <userinput>copy running-config startup-config</userinput>
```

Modify 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 the Flooding 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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **router ospfv3***instance-tag* command to create a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 201
 switch(config-router)#
```

Step 3 Use the **timers lsa-arrival***msec* command to set the LSA arrival time in milliseconds.

Example:

```
switch(config-router)# timers
 lsa-arrival 2000
```

The range is from 10 to 600000. The default is 1000 milliseconds.

Step 4 Use the **timers lsa-group-pacing***seconds* command to set the interval in seconds for grouping LSAs.

Example:

```
switch(config-router)# timers
 lsa-group-pacing 200
```

The range is from 1 to 1800. The default is 10 seconds.

Step 5 Use the **timers throttle lsa***start-time hold-interval max-time* command to set the rate limit in milliseconds for generating LSAs.

Example:

```
switch(config-router)# timers
 throttle lsa network 350 5000 6000
```

You can configure theses timers:

- *start-time*—The range is from 0 to 5000 milliseconds. The default value is 0 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 Use the **address-family ipv6 unicast** command to enter IPv6 unicast address family mode.

Example:

```
switch(config-router)# address-family
 ipv6 unicast
 switch(config-router-af)#
```

Step 7 Use the **timers throttle spf***delay-time hold-time max-time* command to set the SPF best path schedule.

Example:

```
switch(config-router-af)# timers throttle
 spf 3000 2000
```

Sets the SPF best path schedule in seconds between SPF best path calculations with these timers:

- *delay-time*—The range is from 1 to 600,000 milliseconds. The default value is 200 milliseconds.
- *hold-time*—The range is from 1 to 600,000 milliseconds. The default value is 1000 milliseconds.
- *max-wait* —The range is from 1 to 600,000 milliseconds. The default value is 5000 milliseconds.

Step 8 Use the **interface***type slot/port* command to enter interface configuration mode.

Example:

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

Step 9 Use the **ospfv3 retransmit-interval***seconds* command to set the estimated time in seconds between LSAs transmitted from this interface.

Example:

```
switch(config-if)# ospfv3
 retransmit-interval 30
```

The range is from 1 to 65535. The default is 5.

Step 10 Use the **ospfv3 transmit-delay***seconds* command to set the estimated time in seconds to transmit an LSA to a neighbor.

Example:

```
switch(config-if)# ospfv3
 transmit-delay 600
```

The range is from 1 to 450. The default is 1.

Step 11 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

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

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router ospfv3 201</userinput>
switch(config-router)# <userinput>timers lsa-group-pacing 300</userinput>
switch(config-router)# <userinput>copy running-config startup-config</userinput>
```

Configure 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 section).

- Ensure that all neighbors are configured for graceful restart with matching optional parameters set.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **router ospfv3***instance-tag* command to create a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 201
switch(config-router)#
```

Step 3 Use the **graceful-restart** command to enable a graceful restart. A graceful restart is enabled by default.

Example:

```
switch(config-router)# graceful-restart
```

Step 4 Use the **graceful-restart grace-period***seconds* command to set the grace period, in seconds.

Example:

```
switch(config-router)# graceful-restart
grace-period 120
```

The range is from 5 to 1800 seconds. The default is 60 seconds.

Step 5 Use the **graceful-restart helper-disable** command to disable helper mode.

Example:

```
switch(config-router)# graceful-restart
helper-disable
```

It's enabled by default.

Step 6 Use the **graceful-restart planned-only** command to configure graceful restart for planned restarts only.

Example:

```
switch(config-router)# graceful-restart
planned-only
```

Step 7 (Optional) Use the **show ipv6 ospfv3***instance-tag* command to see OPSFv3 information.

Example:

```
switch(config-router)# show ipv6 ospfv3
201
```

Step 8 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

This example shows how to enable a graceful restart if it has been disabled and set the grace period to 120 seconds:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router ospfv3 201</userinput>
switch(config-router)# <userinput>graceful restart</userinput>
switch(config-router)# <userinput>graceful-restart grace-period 120</userinput>
switch(config-router)# <userinput>copy running-config startup-config</userinput>
```

Restart 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

Use the **restart ospfv3** *instance-tag* command to restart the OSPFv3 instance and removes all neighbors.

Example:

```
switch(config)# restart ospfv3 201
```

Configure OSPFv3 with virtualization

You can configure multiple OSPFv3 instances. You can also create multiple VRFs within the virtual device context (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

You must enable OSPFv3 (see the Enabling OSPFv3 section).

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **vrf context***vrf-name* command to create a new VRF and enters VRF configuration mode.

Example:

```
switch(config)# vrf context
RemoteOfficeVRF
switch(config-vrf)#
```

Step 3 Use the **router ospfv3***instance-tag* command to create a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 201
switch(config-router)#
```

Step 4 Use the **vrf***vrf-name* command to enter router VRF configuration mode.

Example:

```
switch(config-router)# vrf
RemoteOfficeVRF
switch(config-router-vrf)#
```

Step 5 (Optional) Use the **maximum-paths***paths* command to configure the maximum number of equal OSPFv3 paths to a destination in the route table for this VRF.

Example:

```
switch(config-router-vrf)# maximum-paths
4
```

Use this command for load balancing.

Step 6 Use the **interface***type slot/port* command to enter interface configuration mode.

Example:

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

Step 7 Use the **vrf member***vrf-name* command to add this interface to a VRF.

Example:

```
switch(config-if)# vrf member
RemoteOfficeVRF
```

Step 8 Use the **ipv6 address***ipv6-prefix/length* command to configure an IP address for this interface.

Example:

```
switch(config-if)# ipv6 address
2001:0DB8::1/48
```

You must do this step after you assign this interface to a VRF.

Step 9 Use the **ipv6 ospfv3***instance-tag***area***area-id* command to assign this interface to the OSPFv3 instance and area configured.

Example:

```
switch(config-if)# ipv6 ospfv3 201
area 0
```

Step 10 (Optional) Use the **copy running-config startup-config** command to copy the running configuration to the startup configuration.

Example:

```
switch(config)# copy running-config startup-config
```

Example

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

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>vrf context NewVRF</userinput>
switch(config-vrf)# <userinput>exit</userinput>
switch(config)# <userinput>router ospfv3 201</userinput>
switch(config-router)# <userinput>exit</userinput>
switch(config)# <userinput>interface ethernet 1/2</userinput>
switch(config-if)# <userinput>vrf member NewVRF</userinput>
switch(config-if)# <userinput>ipv6 address 2001:0DB8::1/48</userinput>
switch(config-if)# <userinput>ipv6 ospfv3 201 area 0</userinput>
switch(config-if)# <userinput>copy running-config startup-config</userinput>
```

Configure Encryption and Authentication

Beginning with Cisco Nexus Release 10.2(1), you can encrypt and authenticate OSPFv3 messages using ESP encapsulation. OSPFv3 depends on IPSec for secure connection. IPSec supports two encapsulation types: Authentication Header (AH) and Encapsulating Security Payload (ESP). RFC4552 'Authentication/Confidentiality for OSPFv3' covers both aspects. ESP configuration provides both encryption and authentication for OSPFv3 messages.

Beginning with Cisco Nexus Release 10.4(1)F, the encryption and authentication algorithms and keys can be configured using the keychain option.

Procedure

Step 1 The following are the limitations:

- Only IPSec transport mode is supported and tunnel mode is not supported.
- AH and ESP configurations together are not allowed on an interface. Though two different interfaces can have AH and ESP.
- Non-disruptive rekeying as defined in section 10 of RFC 4552 is not supported.
- The following Encryption Algorithms will be supported under ESP:
  - AES-CBC (128 bit)
  - AES 192 bit and AES 256 bit will not be supported in this release.
  - 3DES-CBC
  - NULL

- The following Authentications will be supported under ESP:
  - SHA-1
  - NULL
- Both Encryption and Authentication algorithms cannot be configured NULL in one ESP CLI.
- An interface which is part of multiple areas uses the same ESP parameters as the parent.
- On SPI conflict during configuration, error will be thrown to user and configuration will not be saved. So, while changing the ESP configuration the user must use different SPI for a new configuration.
- Max 128 SA/SPI values can be configured per OSPFv3 process.

Step 2 You can configure ESP at the following levels:

- Router
- Area
- Interface
- Virtual Links

Configuring OSPFv3 encryption at router level

You can configure OSPFv3 ESP to encrypt and authenticate OSPFv3 packets at the router level using the following commands.

Before you begin

Enable OSPFv3 feature.

Enable authentication package.

Procedure

Step 1 Enter the global configuration mode:

switch# **configure terminal**

Step 2 Enable OSPFv3:

switch(config)# **feature ospfv3**

Step 3 Enable authentication package:

switch(config)# **feature imp**

Step 4 Create a new OSPFv3 instance with the configured instance tag:

switch(config)# **router ospfv3** *instance-tag*

Step 5 Enable IPSec ESP encryption:

switch(config-router)# **encryption ipsec spi** *spi_id* **esp** [*encrypt_algorithm* [**0** | **3** | **7**] *key* | **key-chain** *enc_keychain_name* | **null**] **authentication** [*auth_algorithm* [**0** | **3** | **7**] *key* | **key-chain** *auth_keychain_name* | **null**]

You can specify the security policy index through *spi_id* and define the encryption algorithm through *encrypt_algorithm* which can be 3DES, AES 128 or null. Numbers 0, 3, and 7 specify the format of the *key*. You can define the authentication algorithm through *auth_algorithm* which can be SHA-1 or NULL.

You can also configure keys and algorithms using the **key-chain** option.

Step 6 (Optional) Display OSPFv3 information:

switch(config)# **show running-config ospfv3**

Configure OSPFv3 encryption at area level

You can configure OSPFv3 ESP to encrypt and authenticate OSPFv3 packets at the area level using the following commands.

Before you begin

Enable OSPFv3 feature.

Enable authentication package.

Procedure

Step 1 Enter the global configuration mode:

switch# **configure terminal**

Step 2 Enable OSPFv3:

switch(config)# **feature ospfv3**

Step 3 Enable the authentication package:

switch(config)# **feature imp**

Step 4 Create a new OSPFv3 instance with the configured instance tag:

switch(config)# **router ospfv3** *instance-tag*

Step 5 Enable IPSec ESP Encryption:

switch(config-router)#**area** *area-num* **encryption ipsec spi** *spi_val* **esp** *encrypt_algorithm* [**0** | **3** | **7***key* | **key-chain** *enc_keychain_name* | **null**] **authentication** *auth_algorithm* [**0** | **3** | **7**] *key* | **key-chain** *auth_keychain_name* | **null**]

You can specify the security policy index through *spi_id* and define the encryption algorithm through *encrypt_algorithm* which can be 3DES, AES 128 or null. Numbers 0, 3, 6 and 7 specify the format of the *key*. You can define the authentication algorithm through *auth_algorithm* which can be SHA-1 or NULL or key-chain.

You can also configure keys and algorithms using the **key-chain** option.

Step 6 (Optional) Display OSPFv3 information:

switch(config)# **show running-config ospfv3**

Configure OSPFv3 encryption at interface level

You can configure OSPFv3 ESP to encrypt and authenticate OSPFv3 packets at the interface level using the following commands.

Before you begin

You must enable OSPFv3.

- Enable authentication package.

Procedure

Step 1 Enter the global configuration mode:

switch# **configure terminal**

Step 2 Enable OSPFv3:

switch(config)# **feature ospfv3**

Step 3 Enables the authentication mode:

switch(config)# **feature imp**

Step 4 Enters the interface configuration mode:

switch(config)# **interface ethernet** *interface*

Step 5 Specify the OSPFv3 instance and area for the interface:

switch(config-if)# **ipv6 router ospfv3** *instance-tag* **area** *area-id*

Step 6 Enable IPSec ESP Encryption:

switch(config-if)# **ospfv3 encryption ipsec spi** *spi_id* **esp** *encrypt_algorithm* [**0** | **3** | **7**] *key* | **key-chain** *enc_keychain_name* | **null**] **authentication** *auth_algorithm* [**0** | **3** | **7**] *key* | **key-chain** *auth_keychain_name* | **null**]

You can specify the security policy index through *spi_id* and define the encryption algorithm through *encrypt_algorithm* which can be 3DES, AES 128 or null. Numbers 0, 3 and 7 specify the format of the *key*. You can define the authentication algorithm through *auth_algorithm* which can be SHA-1 or NULL.

You can also configure keys and algorithms using the key-chain option.

Step 7 (Optional) Display the running configuration on the interface:

switch(config-if)#**show run interface** *interface*

Example

The following example shows how to enable security for Ethernet interface 3/2:

```
switch# configure terminal
switch(config)# feature ospfv3
switch(config)# feature imp
switch(config)# interface ethernet 3/2
switch(config-if)# ipv6 router ospfv3 1 area 0.0.0.0
switch(config-if)# ospfv3 encryption ipsec spi 444
esp  Specify encryption parameters
switch(config-if)# ospfv3 encryption ipsec spi 444 esp
3des  Use the triple DES algorithim
aes   Use the AES algorithim
<!--NXOS1-307-->key-chain  Encryption password key-chain
null  Use NULL authentication
switch(config-if)# ospfv3 encryption ipsec spi 444 esp aes
128  Use the 128-bit AES algorithim
switch(config-if)# ospfv3 encryption ipsec spi 444 esp aes 128
0     Specifies an UNENCRYPTED encryption key will follow
3     Specifies an 3DES ENCRYPTED encryption key will follow
7     Specifies a Cisco type 7  ENCRYPTED encryption key will follow
WORD  The UNENCRYPTED (cleartext) encryption key
switch(config-if)# ospfv3 encryption ipsec spi 444 esp aes 128
12345678123456781234567812345678 authentication null
switch(config-if)# sh ospfv3 interface
Ethernet3/2 is up, line protocol is up
IPv6 address 1:1:1:1::2/64
Process ID 1 VRF default, Instance ID 0, area 0.0.0.0
Enabled by interface configuration
State DOWN, Network type BROADCAST, cost 40
ESP Encryption AES, Authentication NULL, SPI 444, ConnId 444
switch(config-if)#
```

Configure OSPFv3 encryption for virtual links

You can configure OSPFv3 ESP to encrypt and authenticate OSPFv3 packets for virtual links using the following commands.

Before you begin

Enable OSPFv3 feature.

Enable authentication package.

Procedure

Step 1 Enter the global configuration mode:

switch# **configure terminal**

Step 2 Enable OSPFv3:

switch(config)# **feature ospfv3**

Step 3 Enable the authentication package:

switch(config)# **feature imp**

Step 4 Create a new OSPFv3 instance with the configured instance tag:

switch(config)#**router ospfv3** *instance-tag*

Step 5 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.

switch(config-router)# **area** *area-id* **virtual-link** *router-id*

Step 6 Enable IPSec ESP Encryption:

switch(config-router-vlink)# **encryption ipsec spi** *spi_id* **esp** *encrypt_algorithm* [**0** | **3** | **7**] *key* | **key-chain** *enc_keychain_name* | **null**] **authentication** *auth_algorithm* [**0** | **3** | **7**] *key* | **key-chain** *auth_keychain_name* | **null**]

You can specify the security policy index through *spi_id* and define the encryption algorithm through *encrypt_algorithm* which can be 3DES, AES 128 or null. Numbers 0, 3 and 7 specify the format of the *key*. You can define the authentication algorithm through *auth_algorithm* which can be SHA-1 or NULL.

You can also configure keys and algorithms using the **key-chain** option.

Step 7 (Optional) Display OSPFv3 information:

switch(config)# **show running-config ospfv3**

Configuration Example

The following example shows how to encrypt Virtual links:

```
switch(config)# feature ospfv3
switch(config)# feature imp
switch(config-if)# router ospfv3 1
switch(config-router)# area 0.0.0.1 virtual-link 3.3.3.3
switch(config-router-vlink)# encryption ipsec spi ?
<256-4294967295> SPI Value
switch(config-router-vlink)# encryption ipsec spi 256 esp ?
3des Use the triple DES algorithim
aes Use the AES algorithim
key-chain  Encryption password key-chain
null Use NULL authentication
switch(config-router-vlink)# encryption ipsec spi 256 esp aes 128
123456789A123456789B123456789C12 authentication ?
null Use NULL authentication
sha1 Use the SHA1 algorithim
switch(config-router-vlink)# encryption ipsec spi 256 esp aes 128
123456789A123456789B123456789C12 authentication null
```

Note To permit multiple OSPFv3 neighbors to have IPsec ESP, the following policy-map has to be applied for a control-plane:

```
ipv6 access-list copp-acl-ipsec
10 permit ahp any any
20 permit esp any any

class-map type control-plane match-any copp-class-critical-customized-copp
match access-group name copp-acl-ipsec
policy-map type control-plane customized-copp
class copp-class-critical-customized-copp
police cir 36000 kbps bc 1280000 bytes conform transmit violate drop
control-plane
service-policy input customized-copp
```

Configure OSPFv3 authentication at router level

You can configure OSPFv3 ESP to authenticate OSPFv3 packets at the router level using the following commands.

Before you begin

Ensure that you have enabled OSPFv3, for more information see Enabling OSPFv3 section.

Procedure

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 **feature imp**

Example:

```
switch(config)# feature imp
```

Enables authentication mode.

Step 4 **router ospfv3 instance-tag**

Example:

```
switch(config)# router ospfv3 100
switch(config-router)#
```

Creates a new OSPFv3 instance with the configured instance tag.

Step 5 [**no**] **authentication** {**ipsec spi** *spi_id* [*auth_algorithm* [**0** | **3** | **7**] *key* | **key-chain** *auth_keychain_name* | **null**]

Example:

For authentication algorithm and key option:

```
switch(config-router)# authentication ipsec spi 475 md5 11111111111111122222222222222222
```

For keychain option:

```
switch(config-router)# authentication ipsec spi 333 key-chain test1
```

Configures OSPFv3 IPSec authentication at the process (or VRF) level.

The spi argument specifies the security parameter index (SPI). The range is from 256 to 4294967295.

The auth argument specifies the type of authentication. The supported values are md5 or sha1.

0 configures the password in cleartext. 3 configures the pass key as 3DES encrypted. 7 configures the key as Cisco type 7 encrypted.

If the cleartext option (0) is used, the key argument must be 32 characters long for md5 or 40 characters long for sha1.

Beginning with Cisco NX-OS Release 10.4(1)F, the **key-chain** option is provided to configure key and algorithm.

Use the **no** form of this command to disable the OSPFv3 IPSec authentication.

Step 6 (Optional) **show running-config ospfv3**

Example:

```
switch(config)# show running-config ospfv3
```

Displays the OSPFv3 authentication configuration information.

Step 7 (Optional) **copy running-config startup-config**

Example:

```
switch(config)# copy running-config
startup-config
```

Saves this configuration change.

Configure OSPFv3 authentication at area level

You can configure OSPFv3 ESP to authenticate OSPFv3 packets at the area level using the following commands.

For information on how to configure a keychain, see **Configuring Keychain Management** of *Cisco Nexus 9000 Series NX-OS Security Configuration Guide*.

Before you begin

Ensure that you have enabled OSPFv3, for more information see Enabling OSPFv3 section.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **feature ospfv3** command to enable OSPFv3.

Example:

```
switch(config)# feature ospfv3
```

Step 3 Use the **feature imp** command to enable authentication mode.

Example:

```
switch(config)# feature imp
```

Step 4 Use the **router ospfv3 instance-tag** command to create a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 100
                        switch(config-router)#
```

Step 5 Use the [**no**] **area** *area-id-ip* **authentication** {**ipsec spi** *spi_id*[*auth_algorithm* [**0** | **3** | **7**] *key* | **key-chain** *auth_keychain_name* | **null**] command to configure OSPFv3 IPSec authentication at the area level.

Example:

For authentication algorithm and key option:

```
switch(config-router)# area 0 authentication ipsec spi 475 md5 11111111111111112222222222222222
```

For keychain option:

```
switch(config-router)# area 0 authentication ipsec spi 333 key-chain test1
```

The spi argument specifies the security parameter index (SPI). The range is from 256 to 4294967295.

The auth argument specifies the type of authentication. The supported values are MD5 or SHA-1.

0 configures the password in cleartext. 3 configures the pass key as 3DES encrypted. 7 configures the key as Cisco Type-7 encrypted.

If the cleartext option (0) is used, the key argument must be 32 characters long for MD5 or 40 characters long for SHA-1.

Beginning with Cisco NX-OS Release 10.4(1)F, the **key-chain** option is provided to configure key and algorithm.

Use the **no** form of this command to disable the OSPFv3 IPSec authentication.

Step 6 Use the **show running-config ospfv3** command to see the OSPFv3 authentication configuration information.

Example:

```
switch(config)# show running-config ospfv3
```

Step 7 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Configure OSPFv3 authentication at interface level

You can configure OSPFv3 ESP to authenticate OSPFv3 packets at interval level using the following commands.

For information on how to configure a keychain, see **Configuring Keychain Management** of *Cisco Nexus 9000 Series NX-OS Security Configuration Guide*.

Before you begin

Ensure that you have enabled OSPFv3, for more information see Enabling OSPFv3 section.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Use the **interface***interface-type slot/port* command to enter interface configuration mode.

Example:

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

Step 3 Use the [**no**] **ospfv3 authentication** {**disable** | **ipsec spi** *spi_id* {**md5** *akey* | **sha1** *akey* | **key-chain** *keychain_ah*}} command to configure OSPFv3 IPSec authentication for the specified interface.

Example:

For authentication algorithm and key option:

```
switch(config-if)# ospfv3 authentication ipsec spi 475 md5 11111111111111112222222222222222
```

For keychain option:

```
switch(config-if)# ospfv3 authentication ipsec spi 333 key-chain test1
```

The spi argument specifies the security parameter index (SPI). The range is from 256 to 4294967295.

The auth argument specifies the type of authentication. The supported values are MD5 or SHA-1.

0 configures the password in cleartext. 3 configures the pass key as 3DES encrypted. 7 configures the key as Cisco Type-7 encrypted.

If the cleartext option (0) is used, the key argument must be 32 characters long for MD5 or 40 characters long for SHA-1.

Beginning with Cisco NX-OS Release 10.4(1)F, the **key-chain** option is provided to configure key and algorithm.

Use the **no** form of this command to disable the OSPFv3 IPSec authentication.

Step 4 Use the **show running-config ospfv3** command to see the OSPFv3 authentication configuration information.

Example:

```
switch(config)# show running-config ospfv3
```

Step 5 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Configure OSPFv3 authentication at virtual links level

You can configure OSPFv3 ESP to authenticate OSPFv3 packets at the virtual link level using the following commands.

For information on how to configure a keychain, see **Configuring Keychain Management** of *Cisco Nexus 9000 Series NX-OS Security Configuration Guide*.

Before you begin

Ensure that you have enabled OSPFv3, for more information see Enabling OSPFv3 section.

Procedure

Step 1 Use the **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 Use the **feature ospfv3** command to enable OSPFv3.

Example:

```
switch(config)# feature ospfv3
```

Step 3 Use the **feature imp** command to enable authentication mode.

Example:

```
switch(config)# feature imp
```

Step 4 Use the **router ospfv3 instance-tag** command to create a new OSPFv3 instance with the configured instance tag.

Example:

```
switch(config)# router ospfv3 100
                      switch(config-router)#
```

Step 5 Use the **area** *area-id* **virtual-link** *router-id* command to create one end of a virtual link to a remote router.

Example:

```
switch(config-router)# area 0.0.0.10 virtual-link 2001:0DB8::1
                      switch(config-router-vlink)#
```

You must create the virtual link on that remote router to complete the link.

Step 6 Use the [**no**] **authentication** {**ipsec spi** *spi_id* [*auth_algorithm* [**0** | **3** | **7**] *key* | **key-chain** *auth_keychain_name* | **null**] command to configure OSPFv3 IPSec authentication at the virtual link level.

Example:

For authentication algorithm and key option:

```
switch(config-router-vlink)# authentication ipsec spi 475 md5 11111111111111112222222222222222
```

For keychain option:

```
switch(config-router-vlink)# authentication ipsec spi 333 key-chain test1
```

The spi argument specifies the security parameter index (SPI). The range is from 256 to 4294967295.

The auth argument specifies the type of authentication. The supported values are MD5 or SHA-1.

0 configures the password in cleartext. 3 configures the pass key as 3DES encrypted. 7 configures the key as Cisco Type-7 encrypted.

If the cleartext option (0) is used, the key argument must be 32 characters long for MD5 or 40 characters long for SHA-1.

Beginning with Cisco NX-OS Release 10.4(1)F, the **key-chain** option is provided to configure key and algorithm.

Use the **no** form of this command to disable the OSPFv3 IPSec authentication.

Step 7 (Optional) Use the **show running-config ospfv3** command to see the OSPFv3 authentication configuration information.

Example:

```
switch(config)# show running-config ospfv3
```

Step 8 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config startup-config
```

Verify the OSPFv3 configuration

To display the OSPFv3 configuration, perform one of the following tasks:

Command	Purpose
show ipv6 ospfv3 [instance-tag] [vrf vrf-name]	Displays information about one or more OSPFv3 routing instances. The output includes the following area-level counts: • Interfaces in this area—A count of all interfaces added to this area (configured interfaces). • Active interfaces—A count of all interfaces considered to be in router link states and SPF (UP interfaces). • Passive interfaces—A count of all interfaces considered to be OSPF passive (no adjacencies will be formed). • Loopback interfaces—A count of all local loopback interfaces.

Command	Purpose
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 information.
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.
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.

Monitor 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:

```
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
```

This example shows how to configure OSPFv3 encryption using **key-chain** option:

```
switch(config-if)# ospfv3 encryption ipsec spi 333 esp ?
        3des       Use the triple DES algorithim
        aes        Use the AES algorithim
        key-chain  Encryption password key-chain
        null       Use NULL authentication
        switch(config-if)# ospfv3 encryption ipsec spi 333 esp key-chain ?
        WORD  Encryption key-chain name (Max Size 63)
        switch(config-if)# ospfv3 encryption ipsec spi 333 esp key-chain test1 ?
        authentication  Specify authentication parameters
       switch(config-if)# ospfv3 encryption ipsec spi 333 esp key-chain test1 authentication
 ?
        key-chain  Authentication password key-chain
        null       Use NULL authentication
       switch(config-if)# ospfv3 encryption ipsec spi 333 esp key-chain test1 authentication
 key-chain ?
        WORD Authentication key-chain name (Max Size 63)
      switch(config-if)# ospfv3 encryption ipsec spi 333 esp key-chain test1 authentication
 key-chain test2 ?
        <CR>
        switch(config-router)# sh ospfv3
        Routing Process 2 with ID 20.20.10.2 VRF default
        Routing Process Instance Number 1
        Install discard route for summarized internal routes.
        ESP Encryption 3DES, Authentication SHA1, SPI 334, ConnId 334
        ESP keychains: Encr test_key_chain_01(ready), Auth test1(ready)
        Number of new LSAs originated : 3
        Number of new LSAs received : 0
```

Related topics

The following topics can give more information on OSPF:

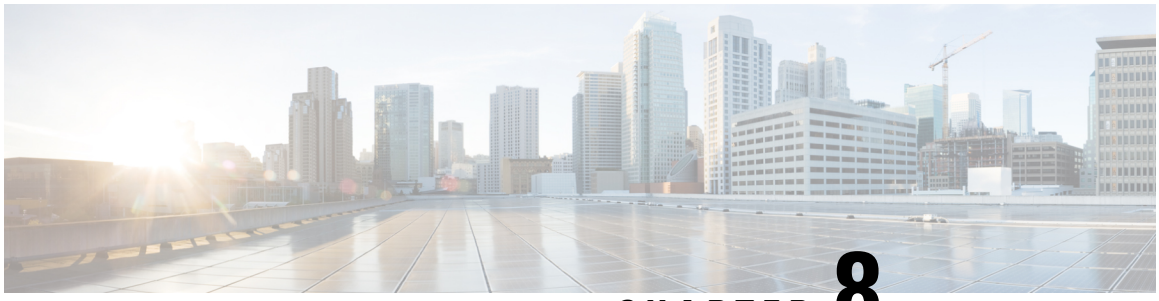
- OSPFv2, on page 111
- Configuring Route Policy Manager, on page 523

Additional references

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

MIBs

MIBs	MIBs Link
MIBs related to OSPFv3	To locate and download supported MIBs, go to the following URL: https://www.cisco.com/c/en/us/td/docs/switches/datacenter/sw/mib/quickreference/cisco-nexus-7000-series-and-9000-series-nx-os-mib-quick-reference.html#con_67262



CHAPTER 8

EIGRP

This chapter describes how to configure the Enhanced Interior Gateway Routing Protocol (EIGRP) on the Cisco NX-OS device.

- EIGRP, on page 215
- Prerequisites for EIGRP, on page 222
- Guidelines and limitations for EIGRP, on page 223
- Default settings, on page 224
- Configuring Basic EIGRP, on page 225
- Configure Advanced EIGRP, on page 230
- Configure Virtualization for EIGRP, on page 245
- Verify the EIGRP Configuration, on page 247
- Monitor EIGRP, on page 247
- Configuration Examples for EIGRP, on page 248
- Related Topics, on page 249
- Additional References, on page 249
- Related Documents, on page 249
- MIBs, on page 249

EIGRP

Enhanced Interior Gateway Routing Protocol (EIGRP) is a routing protocol that 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 theses components:

- Reliable Transport Protocol
- Neighbor Discovery and Recovery

- Diffusing Update Algorithm

Reliable transport protocol

Reliable Transport Protocol is a communication protocol that

- guarantees ordered delivery of EIGRP packets to all neighbors. See the Neighbor Discovery and Recovery
- supports an intermixed transmission of multicast and unicast packets, and
- keeps convergence time remains low for various speed links by sending multicast packets quickly when unacknowledged packets are pending.

See the Configure Advanced EIGRP, on page 230 section for details about modifying the default timers that control the multicast and unicast packet transmissions.

The Reliable Transport Protocol includes theses 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.
- Acknowledgment: Verify reliable reception of Updates, Queries, and Replies.
- Updates: Send 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 part of the Diffusing Update Algorithm used by EIGRP.

Neighbor discovery and recovery

Neighbor discoveries and recoveries are processes used by the Enhanced Interior Gateway Routing Protocol (EIGRP) to identify, maintain, and recover communication with directly connected EIGRP routers. These processes ensure that routers exchange routing information efficiently and maintain accurate knowledge of network topology.

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.

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 these information:

- IPv4 or IPv6 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.

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 these 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. The delay is 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 Cisco recommends 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.

Wide metrics

Wide metrics in EIGRP are 64-bit metric values used to improve route selection on higher-speed or bundled interfaces. They provide more precise path cost calculations compared to traditional 32-bit metrics, enabling better routing decisions in modern networks.

Routers supporting wide metrics can interoperate with routers that do not support wide metrics as these:

- A router that supports wide metrics: Adds local wide metrics values to the received values and sends the information on.
- A router that does not support wide metrics: Sends any received metrics on without changing the values.

EIGRP uses these equation to calculate path cost with wide metrics:

- metric = [k1 x bandwidth + (k2 x bandwidth)/(256 – load) + k3 x delay + k6 x extended attributes] x [k5/(reliability + k4)]
- Since the unicast RIB cannot support 64-bit metric values, EIGRP wide metrics uses these equation with a RIB scaling factor to convert the 64-bit metric value to a 32-bit value:
  - RIB Metric = (Wide Metric / RIB scale value)
  - where the RIB scale value is a configurable parameter.

  EIGRP wide metrics introduce these two new metric values represented as k6 in the EIGRP metrics configuration:
  - Jitter: Measured in microseconds and accumulated across all links in the route path.
  - Energy: Measured in watts per kilobit and accumulated across all links in the route path.

EIGRP prefers a path with low or no jitter or energy metric values over a path with higher values.

Note EIGRP wide metrics are sent with a TLV version of 2. For more information, see the Enabling Wide Metrics section.

External route metrics

External routes are routes that occur between neighbors in different EIGRP autonomous systems. These routes have the these 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 both IPv4 and IPv6 address families. For backward compatibility, you can configure EIGRPv4 in route configuration mode or in IPv4 address family mode. You must configure EIGRP for IPv6 in address family mode.

Address family configuration mode includes these 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. See the *Cisco Nexus 9000 Series NX-OS Security Configuration Guide* for more details about creating keychains.

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

Stub routers are routers that connect to an EIGRP network through a remote router and use the EIGRP stub routing feature to enhance network stability, reduce resource usage, and simplify configuration. See Stub Routing for more details.

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 an active condition and cause instability elsewhere in the network. EIGRP stub routing allows you to prevent queries to the remote router.

Route summarization

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.

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.

In case of process restart or system switchover, the summary address can cause traffic loss. The traffic loss will be seen on the PEER where traffic is routed using the summary address.



Note EIGRP does not support automatic route summarization.

Route redistribution

Route redistribution is a processes that allows EIGRP to learn routes from other routing protocols. You can configure EIGRP to assign a specific link cost to these redistributed routes or apply a default link cost to all of them.

You can use EIGRP to redistribute static routes, routes learned by other EIGRP autonomous systems, or routes from other protocols. You must configure a 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, on page 523.

You also configure the default metric that is used for all imported routes into EIGRP.

You use distribute lists to filter routes from routing updates. These filtered routes are applied to each interface with the **ip distribute-list eigrp** command.

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 usage 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

Split horizon is a routing feature that ensures EIGRP never advertises a route out of the interface where it was learned.

Split horizon 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 the interface from which EIGRP learned the route.

EIGRP uses split horizon or split horizon with poison reverse in these 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.

BFD

Bidirectional forwarding detection (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.

EIGRP supports BFD. See the Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide for more information.

Virtualization support

EIGRP supports virtual routing and forwarding (VRFs) instances.

Graceful restart and high availability

Cisco NX-OS supports nonstop forwarding and graceful restart for EIGRP.

You can use nonstop forwarding for EIGRP to forward data packets along known routes in the FIB while the EIGRP routing protocol information is being restored following a failover. With nonstop forwarding (NSF), peer networking devices do not experience routing flaps. During failover, data traffic is forwarded through intelligent modules while the standby supervisor becomes active.

If a Cisco NX-OS system experiences a cold reboot, the device does not forward traffic to the system and removes the system from the network topology. In this scenario, EIGRP experiences a stateless restart, and all neighbors are removed. Cisco NX-OS applies the startup configuration, and EIGRP rediscovers the neighbors and shares the full EIGRP routing information again.

A dual-supervisor platform that runs Cisco NX-OS can experience a stateful supervisor switchover. Before the switchover occurs, EIGRP uses a graceful restart to announce that EIGRP will be unavailable for some time. During a switchover, EIGRP uses nonstop forwarding to continue forwarding traffic based on the information in the FIB, and the system is not taken out of the network topology.

The graceful restart-capable router uses Hello messages to notify its neighbors that a graceful restart operation has started. When a graceful restart-aware router receives a notification from a graceful restart-capable neighbor that a graceful restart operation is in progress, both routers immediately exchange their topology tables. The graceful restart-aware router performs these actions to assist the router restart:

- The router expires the EIGRP Hello hold timer to reduce the time interval set for Hello messages. This process allows the graceful restart-aware router to reply to the restarting router more quickly and reduces the amount of time required for the restarting router to rediscover neighbors and rebuild the topology table.
- The router starts the route-hold timer. This timer sets the period of time that the graceful restart-aware router will hold known routes for the restarting neighbor. The default time period is 240 seconds.
- The router notes in the peer list that the neighbor is restarting, maintains adjacency, and holds known routes for the restarting neighbor until the neighbor signals that it is ready for the graceful restart-aware router to send its topology table or the route-hold timer expires. If the route-hold timer expires on the graceful restart-aware router, the graceful restart-aware router discards held routes and treats the restarting router as a new router that joins the network and reestablishes adjacency.

After the switchover, Cisco NX-OS applies the running configuration, and EIGRP informs the neighbors that it is operational again.

Multiple EIGRP instances

Cisco NX-OS supports multiple instances of the EIGRP protocol that run on the same system. Every instance uses the same system router ID. You can optionally configure a unique router ID for each instance. For the number of supported EIGRP instances, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

Prerequisites for EIGRP

EIGRP has these prerequisites:

- You must enable EIGRP. See the Enabling the EIGRP Feature for more details.

Guidelines and limitations for EIGRP

EIGRP has these configuration guidelines and limitations:

- When you configure a table map, administrative distance of the routes and the metric, the configuration commands cause the EIGRP neighbors to flap. This is an expected behavior.
- Names in the prefix-list are case-insensitive. We recommend using unique names. Do not use the same name by modifying uppercase and lowercase characters. For example, CTCPrimaryNetworks and CtcPrimaryNetworks are not two different entries.
- 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, on page 523.
- 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, 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 devices 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.
- EIGRP is not supported in tunnel interfaces.
- 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.
- A mix of standard metrics and wide metrics in an EIGRP network with interface speeds of 1 Gigabit or greater might result in suboptimal routing.
- Consider using stubs for larger networks.
- Until NX-OS Release 10.3(3)F, EIGRP redistribute maximum-prefix feature is enabled by default with a default limit of 10000 (10K). Later releases don't have this configuration enabled by default, and an upgrade from NX-OS Release 10.3(3)F will remove this default configuration even if a user has explicitly configured the maximum-prefix value as 10000. Only when a user configures a limit other than 10000, the configuration will be present after upgrading from 10.3(3)F.
- Avoid redistribution between different EIGRP autonomous systems because the EIGRP vector metric will not be preserved.
- The **no** { **ip** | **ipv6** } **next-hop-self** command does not guarantee reachability of the next hop.
- The { **ip** | **ipv6** } **passive-interface eigrp** command suppresses neighbors from forming.
- Cisco NX-OS does not support IGRP or connecting IGRP and EIGRP clouds.
- Auto summarization is disabled by default and cannot be enabled.
- Cisco NX-OS supports only IP.

- High availability is not supported with EIGRP aggressive timers.
- To configure non default aggressive hello timers, it is recommended to use BFD with EIGRP default timers.
- Beginning with Cisco NX-OS Release 9.3(4), if the filtered list is modified when redistributing routes into EIGRP and filtering prefixes with a route map or prefix list, all prefixes that are permitted by the filter, even those not touched, are refreshed in the EIGRP topology table. This refresh is signaled to all EIGRP routers in the query domain for this set of prefixes.
- Beginning with Cisco NX-OS Release 10.3(1)F, EIGRP is supported on the Cisco Nexus 9808 platform switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, EIGRP is supported on the Cisco Nexus 9804 platform switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, EIGRP is supported on N9KX98900CD-A and N9KX9836DM-A line cards with Cisco Nexus 9808 and 9804 switches.
- Beginning with Cisco NX-OS Release 10.4(2)F, EIGRP is supported on Cisco Nexus N9K-C9232E-B1 platform switches.
- Beginning with Cisco NX-OS Release 10.6(2)F, EIGRP is supported on Cisco Nexus N9324C-SE1U and N9348Y2C6D-SE1U platform switches.
- With ASCII reload, VRF configuration is added automatically for all the VRFs under EIGRP

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

Default settings

The table lists the default settings for EIGRP parameters.

Table 20: Default settings for EIGRP parameters

Parameters	Default
Administrative distance	• Internal routes—90 • External routes—170
Bandwidth percent	50 percent

Parameters	Default
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	0 1 0 1 0 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

Configuring Basic EIGRP.

Enable the EIGRP Feature

You must enable EIGRP before you can configure EIGRP.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 [**no**] **feature eigrp**

Example:

```
switch(config)# feature eigrp
```

Enables the EIGRP feature.

The **no** option disables the EIGRP feature and removes all associated configurations.

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.

Create 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

You must enable EIGRP (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 remains in the shutdown state. For IPv6, this number must be configured under the address family.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 [**no**] **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.

Use the **no** option with this command to delete 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.

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 Required: **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 Required: {**ip** | **ipv6**} **router eigrp** *instance-tag*

Example:

```
switch(config-if)# ip router eigrp Test1
```

```
R2(config-if)# vrf member eigrp-vrf
Warning: Retain-L3-config is on, deleted and re-added L3 config on interface Ethernet1/8
VRF eigrp-vrf does not exist. Create vrf to make interface Ethernet1/8 operational
R2(config-if)#
R2(config-if)# sh ru eigrp
```

```
!Command: show running-config eigrp
!Running configuration last done at: Thu Aug 25 06:59:31 2022
!Time: Thu Aug 25 06:59:36 2022

version 10.3(1) Bios:version 05.47
feature eigrp

router eigrp 10
vrf eigrp-vrf

interface Ethernet1/8
ip router eigrp 10
```

Associates this interface with the configured EIGRP process. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.



Note On the interfaces where EIGRP process is running and*vrf retain*is configured, in this case, when the**vrf member**is changed on the interface, then the newly created *vrf-name* will also be reflected under the context of EIGRP process.

Step 8 (Optional) **show** {**ip** | **ipv6**} **eigrp interfaces**

Example:

```
switch(config-if)# show ip eigrp
interfaces
```

Displays information about EIGRP interfaces.

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 an EIGRP process and configure an interface for EIGRP:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router eigrp Test1</userinput>
switch(config-router)# <userinput>interface ethernet 1/2</userinput>
switch(config-if)# <userinput>ip router eigrp Test1</userinput>
switch(config-if)# <userinput>no shutdown</userinput>
switch(config-if)# <userinput>copy running-config startup-config</userinput>
```

For more information about other EIGRP parameters, see the Configure Advanced EIGRP, on page 230 section.

Restart an EIGRP Instance

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

To restart an EIGRP instance and remove all associated neighbors, use the following commands in global configuration mode:

Procedure

Step 1 (Optional) **flush-routes**

Example:

```
switch(config)# flush-routes
```

Flushes all EIGRP routes in the unicast RIB when this EIGRP instance restarts.

Step 2 **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 removes all routes and adjacencies but preserves the EIGRP configuration.

To disable an EIGRP instance, use the following command in router configuration mode:

Procedure

shutdown

Example:

```
switch(config-router)# shutdown
```

Disables this instance of EIGRP. The EIGRP router configuration remains.

Configure a Passive Interface for EIGRP

You can configure a passive interface for EIGRP. A passive interface does not participate in EIGRP adjacency, but the network address for the interface remains in the EIGRP topology table.

To configure a passive interface for EIGRP, use the following command in interface configuration mode:

Procedure

{**ip** | **ipv6**} **passive-interface eigrp** *instance-tag*

Example:

```
switch(config-if)# ip passive-interface eigrp tag10
```

Suppresses EIGRP hellos, which prevents neighbors from forming and sending routing updates on an EIGRP interface. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.

Shut Down EIGRP on an Interface

You can gracefully shut down EIGRP on an interface. This action removes all adjacencies and stops EIGRP traffic on this interface but preserves the EIGRP configuration.

To disable EIGRP on an interface, use the following command in interface configuration mode:

Procedure

{**ip** | **ipv6**} **eigrp** *instance-tag* **shutdown**

Example:

```
switch(config-if)# ip eigrp Test1
shutdown
```

Disables EIGRP on this interface. The EIGRP interface configuration remains. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.

Configure Advanced EIGRP

Configure 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. The interface EIGRP authentication configuration overrides the EIGRP process-level authentication configuration.

Before you begin

You must enable EIGRP (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 the keychain for this authentication configuration. For more information, see the Cisco Nexus 9000 Series NX-OS Security Configuration Guide.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

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** | **ipv6**} **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 key-chain***key-chain*

Example:

```
switch(config-router-af)# authentication
key-chain routeKeys
```

Associates a keychain with this EIGRP process for this VRF. The keychain can be any case-sensitive, alphanumeric string up to 63 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 {**ip** | **ipv6**} **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.

Step 8 {**ip** | **ipv6**} **authentication key-chain eigrp** *instance-tagkeychain*

Example:

```
switch(config-if)# ip authentication
key-chain 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 9 {**ip** | **ipv6**} **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 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 MD5 message digest authentication for EIGRP over Ethernet interface 1/2:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router eigrp Test1</userinput>
switch(config-router)# <userinput>exit</userinput>
switch(config)# <userinput>interface ethernet 1/2</userinput>
switch(config-if)# <userinput>ip router eigrp Test1</userinput>
switch(config-if)# <userinput>ip authentication key-chain eigrp Test1 routeKeys</userinput>
switch(config-if)# <userinput>ip authentication mode eigrp Test1 md5</userinput>
switch(config-if)# <userinput>copy running-config startup-config</userinput>
```

Configuring EIGRP Stub Routing

You can configure a router for EIGRP stub routing.

To configure a router for EIGRP stub routing, use the following command in address-family configuration mode:

SUMMARY STEPS

1. **stub** [**direct** | **receive-only** | **redistributed** [**direct**] **leak-map** *map-name*]
2. (Optional) **show ip eigrp neighbor detail**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**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 leak-map map-name name refers to a configured route-map. Multiple options can be configured at once to enable the desired stub router functionality.
Step 2	(Optional) **show ip eigrp neighbor detail** **Example:** `switch(config-router-af)# show ip eigrp neighbor detail`	Verifies that the router has been configured as a stub router.

Example

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 ipv6 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
```

Configure 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 advertises 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:

Procedure

{**ip** | **ipv6**} **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 an IP prefix/length. The instance tag and map name can be any case-sensitive, alphanumeric string up to 20 characters which refers to a configured route-map.

You can optionally configure the administrative distance for this aggregate address. The default administrative distance is 5 for aggregate addresses.

Note We recommend that you configure the IP address using the `prefix/length` format instead of `address mask` unless EIGRP is already running. If you use the `address mask` format before the EIGRP instance has started, you will be unable to remove or alter the summary address later.

Example

This example shows how to cause EIGRP to summarize network 192.0.2.0 out Ethernet 1/2 only:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>interface ethernet 1/2</userinput>
switch(config-if) <userinput>ip summary-address eigrp Test1 192.0.2.0/24</userinput>
```

Redistribute Routes into EIGRP

You can redistribute routes in EIGRP from other routing protocols.

Before you begin

- You must enable EIGRP (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, on page 523.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

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** | **ipv6**} **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** | **isis** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **direct** | **static**} **route-map** *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 Kbps
- delay—100 (10 microsecond units)
- reliability—255
- loading—1
- MTU—1492

Step 6 (Optional) **show** {**ip** | **ipv6**} **eigrp route-map statistics redistribute**

Example:

```
switch(config-router-af)# show ip eigrp
route-map statistics redistribute bgp
```

Displays information about EIGRP route map statistics.

Step 7 (Optional) **copy running-config startup-config**

Example:

```
switch(config-router-af)# copy
running-config startup-config
```

Saves this configuration change.

Example

The following example shows how to redistribute BGP into EIGRP for IPv4:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router eigrp Test1</userinput>
switch(config-router)# <userinput>redistribute bgp 100 route-map BGPFilter</userinput>
switch(config-router)# <userinput>default-metric 500000 30 200 1 1500</userinput>
switch(config-router)# <userinput>copy running-config startup-config</userinput>
```

Limit 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—EIGRP accepts the redistributed routes up to the configured maximum value. By default, EIGRP logs a warning message when a default threshold of 75% is passed and also when maximum limit is reached. You can optionally configure a threshold percentage of the maximum redistributed routes.
- Warning only—Logs a warning message when threshold percentage of set maximum value is passed. However, EIGRP continues to accept the redistributed routes.
- Withdraw—Starts 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.
- Cisco recommends setting the maximum prefix value to 2 times the expected redistributed routes.
- Route redistribute does not support more than 8 redistribute commands. After configuring 8 commands, the new routes are not added to the routing table or dynamic routing database.



Note This task can be configured only in the IPv4 VRF address family configuration mode.

Before you begin

You must enable EIGRP (see the Enabling the EIGRP Feature section).

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 **router eigrp** *instance-tag*

Example:

```
switch(config)# router eigrp Test1
switch(config-router)#
```

Creates a new EIGRP 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 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 distributes. The range is from 1 to 65535. Optionally specifies 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. 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 the **clear ip eigrp redistribution** command if all routes are withdrawn.

Note In EIGRP topology, it is recommended to set the maximum-prefix value to 2 times the expected redistributed routes.

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# c<userinput>onfigure terminal</userinput>
switch(config)# <userinput>router eigrp Test1</userinput>
switch(config-router)# <userinput>redistribute bgp route-map FilterExternalBGP</userinput>
switch(config-router)# <userinput>redistribute maximum-prefix 1000 75</userinput>
```

Configure 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. See the Configuring Load Balancing in EIGRP section.

Before you begin

You must enable EIGRP (see the Enabling the EIGRP Feature section).

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

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** | **ipv6**} **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 accepts in the route table. The range is from 1 to 32. The default is 8.

Step 5 (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 configure equal cost load balancing for EIGRP over IPv4 with a maximum of six equal cost paths:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router eigrp Test1</userinput>
switch(config-router)# <userinput>maximum-paths 6</userinput>
switch(config-router)# <userinput>copy running-config startup-config</userinput>
```

Configure Graceful Restart for EIGRP

You can configure graceful restart or nonstop forwarding for EIGRP. See the Graceful Restart and High Availability section.

Graceful restart is enabled by default.

Before you begin

You must enable EIGRP (see the Enabling the EIGRP Feature section).

- 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.
- Neighboring devices participating in the graceful restart must be NSF aware or NSF capable.

Procedure

Step 1 **configure terminal**

Example:

```
switch# configure terminal
```

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 remains in the shutdown state.

Step 3 **address-family** {**ipv4** | **ipv6**} **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 **graceful-restart**

Example:

```
switch(config-router-af)# graceful-restart
```

Enables graceful restart. This feature is enabled by default.

Step 5 **timers nsf converge***seconds*

Example:

```
switch(config-router-af)# timers nsf converge
100
```

Sets the time limit for the convergence after a switchover. The range is from 60 to 180 seconds. The default is 120.

Step 6 **timers nsf route-hold***seconds*

Example:

```
switch(config-router-af)# timers nsf
route-hold 200
```

Sets the hold time for routes learned from the graceful restart-aware peer. The range is from 20 to 300 seconds. The default is 240.

Step 7 **timers nsf signal***seconds*

Example:

```
switch(config-router-af)# timers nsf signal 15
```

Sets the time limit for signaling a graceful restart. The range is from 10 to 30 seconds. The default is 20.

Step 8 (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 configure graceful restart for EIGRP over IPv6 using the default timer values:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router eigrp Test1</userinput>
switch(config-router)# <userinput>address-family ipv6 unicast</userinput>
switch(config-router-af)# <userinput>graceful-restart</userinput>
switch(config-router-af)# <userinput>copy running-config startup-config</userinput>
```

Adjust 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.

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 step 2 command in interface configuration mode:

Procedure

Step 1 {**ip** | **ipv6**} **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.

Step 2 {**ip** | **ipv6**} **hold-time eigrp** *instance-tag seconds*

Example:

```
switch(config-if)# ipv6 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 seconds.

Example

Use the **show ip eigrp interface detail** command to verify the timer configuration.

Disable 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 devices, 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:

Procedure

no {**ip** | **ipv6**} **split-horizon eigrp** *instance-tag*

Example:

```
switch(config-if)# no ip split horizon eigrp Test1
```

Disables split horizon.

Enable Wide Metrics

To enable wide metrics and optionally configure a scaling factor for the RIB, use the following commands in router or address family configuration mode:

Procedure

Step 1 **metrics version 64bit**

Example:

```
switch(config-router)# metrics version 64bit
```

Enables 64-bit metric values.

Step 2 (Optional) **metrics rib-scale** *value*

Example:

```
switch(config-router)#
```

Configures the scaling factor used to convert the 64-bit metric values to 32 bit in the RIB. The range is from 1 to 255. The default value is 128.

Tune EIGRP

You can configure optional parameters to tune EIGRP for your network.

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

Procedure

Step 1 **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 route-map name can be any case-sensitive, alphanumeric string up to 20 characters.

Step 2 **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).

Step 3 **metric max-hops** *hop-count*

Example:

```
switch(config-router-af)# metric max-hops
70
```

Sets the 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.

Step 4 **metric weights** *tos k1 k2 k3 k4 k5k6*

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 x bandwidth + (k2 x bandwidth)/(256 – load) + k3 x delay + k6 x extended attributes] * [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.
- k6—0. The range is from 0 to 255.

Step 5 **nsf await-redist-proto-convergence**

Example:

```
switch(config-router-af)# nsf
await-redist-proto-convergence
```

Causes EIGRP to wait for the convergence of redistributed protocols before installing its own routes in the Routing Information Base (RIB) during nonstop forwarding (NSF).

This command is useful in switchover scenarios when NSF is in progress and you want EIGRP to wait for BGP to converge and install its routes. It prevents EIGRP from installing transient routes and modifying the Forwarding Information Base (FIB) entries before BGP converges and EIGRP finds an alternate path to a destination.



Note If you use this command when mutual redistribution is configured between EIGRP and BGP (for example, in a PE-CE environment), some traffic loss might occur because the provider-edge (PE) router will not install EIGRP routes into the RIB until BGP routes are available. This behavior delays the routes that the customer-edge (CE) router learns from EIGRP and advertises to the peer PE router.

Step 6 **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.

Step 7 (Optional) {**ip** | **ipv6**} **bandwidth 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 Kbps.

Step 8 {**ip** | **ipv6**} **bandwidth-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.

Step 9 [**no**] {**ip** | **ipv6**} **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).

Step 10 {**ip** | **ipv6**} **distribute-list eigrp** *instance-tag* {**prefix-list** *name* | **route-map** *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.

Step 11 [**no**] {**ip** | **ipv6**} **next-hop-self eigrp** *instance-tag*

Example:

```
switch(config-if)# ipv6 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.

Step 12 {**ip** | **ipv6**} **offset-list eigrp** *instance-tag* {**prefix-list** *name* | **route-map** *map-name*} {**in** | **out**} *offset*

Example:

```
switch(config-if)# ip offset-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.

Step 13 {**ip** | **ipv6**} **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, alphanumeric string up to 20 characters.

Configure Virtualization for EIGRP

You can create multiple VRFs and use the same or multiple EIGRP processes in each VRF. You assign an interface to a VRF.

- 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.

Before you begin

You must enable EIGRP (see the Enabling the EIGRP Feature section).

- Create the VRFs.

Procedure

Step 1 **configure terminal**

Example:

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

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 VRF name can be any case-sensitive, alphanumeric string up to 20 characters.

Step 3 **router eigrp***instance-tag*

Example:

```
switch(config-vrf)# 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 remains 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 **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 6 {**ip** | **ipv6**}**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 7 **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# <userinput>configure terminal</userinput>
switch(config)# <userinput>vrf context NewVRF</userinput>
switch(config-vrf)# <userinput>router eigrp Test1</userinput>
switch(config-router)# <userinput>interface ethernet 1/2</userinput>
switch(config-if)# <userinput>ip router eigrp Test1</userinput>
switch(config-if)# <userinput>vrf member NewVRF</userinput>
switch(config-if)# <userinput>copy running-config startup-config</userinput>
```

Verify the EIGRP Configuration

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

Command	Purpose
show {**ip** | **ipv6**} **eigrp** [*instance-tag*]	Displays a summary of the configured EIGRP processes.
show {**ip** | **ipv6**} **eigrp** [*instance-tag*] **interfaces** [*type number*] [**brief**] [**detail**]	Displays information about all configured EIGRP interfaces.
show {**ip** | **ipv6**} **eigrp** *instance-tag* **neighbors** [*type number*] [**detail**]	Displays information about all the EIGRP neighbors. Use this command to verify the EIGRP neighbor configuration.
show {**ip** | **ipv6**} **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** | **ipv6**} **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.

Monitor EIGRP

To display EIGRP statistics, use the following commands:

Command	Purpose
show {**ip** | **ipv6**} **eigrp** [*instance-tag*] **accounting** [**vrf** *vrf-name*]	Displays accounting statistics for EIGRP.
show {**ip** | **ipv6**} **eigrp** [*instance-tag*] **route-map statistics redistribute**	Displays redistribution statistics for EIGRP.
show {**ip** | **ipv6**} **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
ip address 192.0.2.55/24
ip router eigrp Test1
no shutdown
router eigrp Test1
router-id 192.0.2.1
```

The following example shows how to use a route map with the **distribute-list** command to filter routes that are dynamically received from (or advertised to) EIGRP peers. The example configures a route map to match an EIGRP external protocol metric route with an allowable deviation of 100, a source protocol of BGP, and an autonomous system number of 45000. When the two match clauses are true, the tag value of the destination routing protocol is set to 5. The route map is used to distribute incoming packets for an EIGRP process.

```
switch(config)# route-map metric-range
switch(config-route-map)# match metric external 500 +- 100

switch(config-route-map)# set tag 5
switch(config-route-map)# exit
switch(config)# router eigrp 1
switch(config-router)# exit
switch(config)# interface ethernet 1/2
switch(config-if)# ip address 172.16.0.0
switch(config-if)# ip router eigrp 1
switch(config-if)# ip distribute-list eigrp 1 route-map metric-range in
```

The following example shows how to use a route map with the redistribute command to allow routes that are redistributed from the routing table to be filtered with a route map before being admitted into an EIGRP topology table. The example shows how to configure a route map to match EIGRP routes with a metric of 110, 200, or an inclusive range of 700 to 800. When the match clause is true, the tag value of the destination routing protocol is set to 10. The route map is used to redistribute EIGRP packets.

```
switch(config)# route-map metric-eigrp
switch(config-route-map)# match metric 110 200 750 +- 50
switch(config-route-map)# set tag 10
switch(config-route-map)# exit
switch(config)# router eigrp 1
switch(config-router)# redistribute eigrp route-map metric-eigrp
switch(config-router)# exit
switch(config)# interface ethernet 1/2
switch(config-if)# ip address 172.16.0.0
switch(config-if)# ip router eigrp 1
```

Related Topics

See Configuring Route Policy Manager, on page 523, for more information on route maps.

Additional References

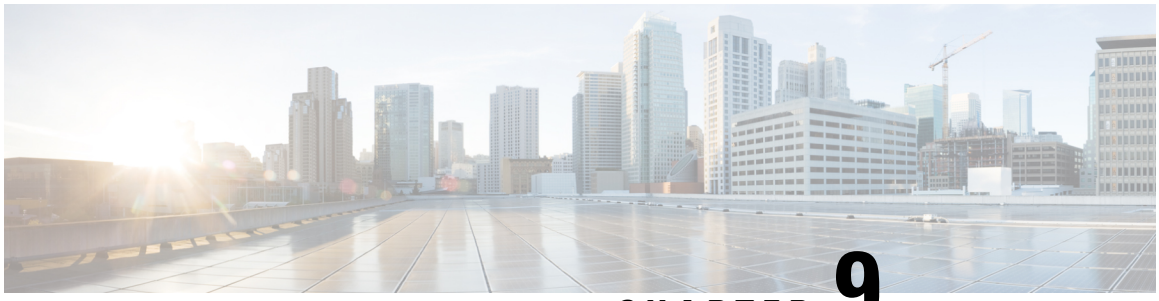
For additional information related to implementing EIGRP, see the following sections:

Related Documents

Related Topic	Document Title
EIGRP CLI commands	*Cisco Nexus 9000 Series NX-OS Unicast Routing Command Reference*
Introduction to EIGRP Tech Note	Introduction to EIGRP Tech Note
EIGRP Frequently Asked Questions	EIGRP Frequently Asked Questions

MIBs

MIBs	MIBs Link
MIBs related to EIGRP	To locate and download MIBs, go to the following URL: https://cisco.github.io/cisco-mibs/supportlists/nexus9000/Nexus9000MIBSupportList.html



CHAPTER 9

Configure 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 theses sections:

- IS-IS, on page 251
- IS-IS authentication, on page 254
- Mesh groups, on page 254
- Overload bits, on page 254
- Route summarization, on page 255
- Route redistribution, on page 255
- Link prefix suppression, on page 255
- Load balance, on page 255
- BFD, on page 256
- Virtualization support, on page 256
- High availability and graceful restart, on page 256
- Multiple IS-IS instances, on page 257
- Prerequisites for IS-IS, on page 257
- Guidelines and limitations for IS-IS, on page 257
- Default settings, on page 257
- Configure IS-IS, on page 258
- Verifying the IS-IS Configuration, on page 283
- Monitor IS-IS, on page 284
- Configuration Examples for IS-IS, on page 285
- Related Topics, on page 285

IS-IS

Intermediate System to Intermediate System (IS-IS) is an Interior Gateway Protocol (IGP) based on Standardization (ISO)/International Engineering Consortium (IEC) 10589 that

- supports Internet Protocol version 4 (IPv4) and IPv6
- is a dynamic link-state routing protocol, and
- can detect changes in the network topology and calculate loop-free routes to other nodes in the network.

Each router running IS-IS 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 hello packets and adjacency formation

Intermediate System to Intermediate System (IS-IS) sends a hello packet out every configured interface to discover IS-IS neighbor routers. The hello packet contains information such as:

- authentication details
- are information
- supported protocols

The receiving interface uses these information to determine compatibility with the originating interface. Additionally, hello packets are padded to ensure that IS-IS only forms adjacencies 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 Configuring the Transient Mode for Hello Padding.

IS-IS areas

An IS-IS area is a network segment within an IS-IS routing domain that organizes routers for efficient routing. IS-IS areas can be designed as a single area encompassing all routers or as multiple areas connected through a backbone, known as the Level 2 area.

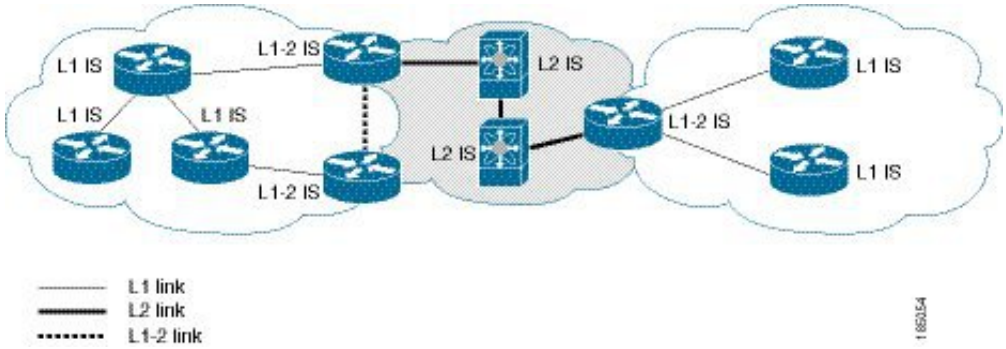
- Level 1 areas: consist of routers called Level 1 routers. These routers establish adjacencies and perform routing only within their local area, known as intra-area routing.
- Level 2 areas: consist of Level 2 routers that establish adjacencies with other Level Level 2 routers and handle routing between Level 1 areas, known as inter-area routing.
- A router can be configured as both Level 1 and Level 2, called a Level 1/Level 2 router. These routers act as area border routers, routing traffic between the local Level 1 area and the Level 2 backbone area. See figure Figure 26: IS-IS Network Divided into Areas, on page 253.

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 26: 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

A Network Entity Title (NET) is a unique identifier associated with each IS-IS (Intermediate System to Intermediate System) instance. It serves to identify the IS-IS instance within an area and consists of two key components:

- System ID: A unique identifier that distinguishes the IS-IS instance within the area.
- Area ID: Identifies the IS-IS area to which the instance belongs.

For example, if the NET is 47.0004.004d.0001.0001.0c11.1111.00, the system ID is 0000.0c11.1111.00 and the area is ID 47.0004.004d.0001.

Designated intermediate systems

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

IS-IS authentication helps you to 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 these 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 use also keychains for hello and LSP authentication. See the Cisco Nexus 9000 Series 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 bits

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 can use 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

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.

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.

Route redistribution

Route redistribution is a processes that allows IS-IS to learn routes from other routing protocols. You can configure IS-IS to assign a specific link cost to these redistributed routes or apply a default link cost to all of them.

You must configure a route map with the redistribution to control which routes are passed into IS-IS. A route map allows you to filter routes based on attributes such as the destination, origination protocol, route type, route tag, and so on. For more information, see Configuring Route Policy Manager, on page 523.

Whenever you redistribute routes into an IS-IS routing domain, Cisco NX-OS does not, by default, redistribute the default route into the IS-IS routing domain. You can generate a default route into IS-IS, which can be controlled by a route policy.

You also configure the default metric that is used for all imported routes into IS-IS.

Link prefix suppression

By default, IS-IS advertises the addresses of connected interfaces in the system LSP. By suppressing the advertisement of unwanted interface addresses, you can reduce the size of LSPs and reduce the number of routes that IS-IS maintains, improving convergence times.

Two prefix suppression methods are provided for reducing the number of routes in the LSP:

- At the global level, you can choose to advertise only those prefixes that belong to passive interfaces, excluding other connected prefixes. See Advertise Only Passive Interface Prefixes, on page 274.
- At the interface level, you can disable the advertisement of connected prefixes. See Suppress Prefixes on an Interface, on page 275.

Load balance

Load balancing is a feature that allows 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 16 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

Bidirectional forwarding detection (BFD) is a detection protocol designed to provide fast forwarding-path failure detection times. IS-IS supports BFD for IPv4 and IPv6.

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 9000 Series 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 9000 Series NX-OS Verified Scalability Guide.

High availability and graceful restart

High availabilities in Cisco NX-OS provide a multilevel architecture that ensures continuous network operation and minimal downtime. IS-IS supports high availability.

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 these scenarios:

- First recovery attempt after process experiences problems
- User-initiated switchover using the **system switchover** command

A graceful restart is used in these 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 9000 Series NX-OS Verified Scalability Guide.

Prerequisites for IS-IS

These are the prerequisites of IS-IS:

- You must enable IS-IS. See the Enabling the IS-IS Feature.

Guidelines and limitations for IS-IS

IS-IS has these configuration guidelines and limitations:

- IS-IS Level-1 routes do not populate on the connecting Level-2-only switch if an explicit configuration is not added to the Level-1/Level-2 Cisco Nexus switch.
- 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.
- You can configure IS-IS over segment routing for all Cisco Nexus 9000 Series switches and the Cisco Nexus 3164Q and 31128PQ switches. For information, see the Cisco Nexus 9000 Series NX-OS Label Switching Configuration Guide.
- Beginning with Cisco NX-OS Release 10.6(1)F, IS-IS is supported on Cisco Nexus N9336C-SE1 switches.

Default settings

The table lists the default settings for IS-IS parameters.

Table 21: Default IS-IS Parameters

Parameters	Default
Administrative distance	115
Area level	Level-1-2
DIS priority	64

Parameters	Default
Graceful restart	Enabled
Hello multiplier	3
Hello padding	Enabled
Hello time	10 seconds
IS-IS feature	Disabled
LSP interval	33
LSP MTU	1492
Maximum LSP lifetime	1200 seconds
Maximum paths	8
Metric	40
Reference bandwidth	40 Gbps

Configure IS-IS

To configure IS-IS, follow these steps:

1. Enable the IS-IS feature (see the Enabling the IS-IS Feature section).
2. Create an IS-IS instance (see the Creating an IS-IS Instance section).
3. Add an interface to the IS-IS instance (see the Configuring IS-IS on an Interface section).
4. Configure optional features, such as authentication, mesh groups, and dynamic host exchange.

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.

IS-IS Configuration Modes

The following sections show how to enter each of the configuration modes. 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)#
```

Enable the IS-IS Feature

You must enable the IS-IS feature before you can configure IS-IS.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 [**no**] **feature isis**

Example:

```
switch(config)# feature isis
```

Enables or disables the IS-IS feature.

Using the **no** option with this command disables the IS-IS feature and removes all associated configurations.

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.

Create 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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 [**no**] **router isis** *instance-tag*

Example:

```
switch(config)# router isis Enterprise
switch(config-router)#
```

Creates a new IS-IS instance with the configured instance tag.

Use the **no** form of this command to delete 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.

Step 3 **net** *network-entity-title*

Example:

```
switch(config-router)# net
47.0004.004d.0001.0001.0c11.1111.00
```

Configures the NET for this IS-IS instance.

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-router)# show isis process
```

Displays a summary of IS-IS information for all IS-IS instances.

Step 6 (Optional) **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.

Step 7 (Optional) **log-adjacency-changes**

Example:

```
switch(config-router)#
log-adjacency-changes
```

Sends a system message whenever an IS-IS neighbor changes the state.

Step 8 (Optional) **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.

Step 9 (Optional) **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.

Step 10 (Optional) **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.

Step 11 (Optional) **clear isis** [*instance-tag*] **adjacency** [***** | *system-id* | *interface*]

Example:

```
switch(config-router)# clear isis adjacency *
```

Clears neighbor statistics and removes adjacencies for this IS-IS instance.

Step 12 (Optional) **copy running-config startup-config**

Example:

```
switch(config-router)# copy running-config
startup-config
```

Saves this configuration change.

Example

The following example shows how to create an IS-IS instance in a level 2 area:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router isis Enterprise</userinput>
switch(config-router)# <userinput>net 47.0004.004d.0001.0001.0c11.1111.00</userinput>
switch(config-router)# <userinput>is-type level-2</userinput>
switch(config-router)# <userinput>copy running-config startup-config</userinput>
```

Restart 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

restart isis *instance-tag*

Example:

```
switch(config)# restart isis Enterprise
```

Restarts the IS-IS instance and removes all neighbors.

Shut 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

shutdown

Example:

```
switch(config-router)# shutdown
```

Disables the IS-IS instance.

Configure IS-IS on an Interface

You can add an interface to an IS-IS instance.

Before you begin

You must enable IS-IS (see the Enabling the IS-IS Feature section).

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 **interface***interface-type slot/port*

Example:

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

Enters interface configuration mode.

Step 3 (Optional) **medium** {**broadcast** | **p2p**}

Example:

```
switch(config-if)# medium p2p
```

Configures the broadcast or point-to-point mode for the interface. IS-IS inherits this mode.

Step 4 {**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 5 (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.

Step 6 (Optional) **isis circuit-type** {**level-1** | **level-2** | **level-1-2**}

Example:

```
switch(config-if)# isis circuit-type
level-2
```

Sets the type of adjacency that this interface participates in. Use this command only for routers that participate in both Level 1 and Level 2 areas.

Step 7 (Optional) **isis metric** *value* {**level-1** | **level-2**}

Example:

```
switch(config-if)# isis metric 30
```

Sets the IS-IS metric for this interface. The range is from 1 to 16777214. The default is 10.

Step 8 (Optional) **isis passive** {**level-1** | **level-2** | **level-1-2**}

Example:

```
switch(config-if)# isis passive level-2
```

Prevents the interface from forming adjacencies but still advertises the prefix associated with the interface.

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 add the Ethernet 1/2 interface to an IS-IS instance:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>interface ethernet 1/2</userinput>
switch(config-if)# <userinput>ip router isis Enterprise</userinput>
switch(config-if)# <userinput>copy running-config startup-config</userinput>
```

Shut Down IS-IS on an Interface

You can gracefully shut down IS-IS on an interface. This action removes all adjacencies and stops IS-IS traffic on this interface but preserves the IS-IS configuration.

To disable IS-IS on an interface, use the following command in interface configuration mode:

Procedure

isis shutdown

Example:

```
switch(config-if)# isis shutdown
```

Disables IS-IS on this interface. The IS-IS interface configuration remains.

Configure 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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

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:

```
switch(config-router)# authentication
key-chain ISISKey level-2
```

Configures the authentication key that is used for an IS-IS area-level authentication.

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# <userinput>configure terminal</userinput>
switch(config)# <userinput>router isis Enterprise</userinput>
```

```
switch(config-router)# <userinput>authentication-type cleartext level-2</userinput>
switch(config-router)# <userinput>authentication key-chain ISISKey level-2</userinput>
switch(config-router)# <userinput>copy running-config startup-config</userinput>
```

Configure 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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

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:

```
switch(config-if)# isis
authentication-type cleartext level-2
```

Sets the authentication type for IS-IS on this interface as cleartext or as an MD5 authentication digest.

Step 4 **isis authentication key-chain** *key* {**level-1** | **level-2**}

Example:

```
switch(config-if)# isis
authentication-key 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# <userinput>configure terminal</userinput>
switch(config)# <userinput>interface ethernet 1/2</userinput>
switch(config-if)# <userinput>isis authentication-type cleartext level-2</userinput>
switch(config-if)# <userinput>isis authentication key-chain ISISKey</userinput>
switch(config-if)# <userinput>copy running-config startup-config</userinput>
```

Configure 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

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.

Configure 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

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:

SUMMARY STEPS

1. **hostname dynamic**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**hostname dynamic** **Example:** `switch(config-router)# hostname dynamic`	Enables dynamic host exchange.

Set 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

set-overload-bit {**always** | **on-startup** {*seconds* | **wait-for bgp** *as-number*}} [**suppress** [**interlevel** | **external**]]

Example:

```
switch(config-router)# set-overload-bit
on-startup 30
```

Sets the overload bit for IS-IS. The *seconds* range is from 5 to 86400.

Configure 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

[no] set-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.

Configure 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 interface configuration mode:

Procedure

[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.

Configure 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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

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:

```
Example:
switch(config-router-af)# 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-router-af)# 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# <userinput>configure terminal</userinput>
switch(config)# <userinput>router isis Enterprise</userinput>
switch(config-router)# <userinput>address-family ipv4 unicast</userinput>
switch(config-router-af)# <userinput>summary-address 192.0.2.0/24 level-2</userinput>
switch(config-router-af)# <userinput>copy running-config startup-config</userinput>
```

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).

SUMMARY STEPS

1. Use the **configure terminal** command to enter global configuration mode.
2. **router isis** *instance-tag*
3. **address-family** {**ipv4** | **ipv6**} **unicast**
4. **redistribute** {**bgp** *as* | {**eigrp** | **isis** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static** | **direct**} **route-map** *map-name*
5. (Optional) **default-information originate** [**always**] [**route-map** *map-name*]
6. (Optional) **distribute** {**level-1** | **level-2**} **into** {**level-1** | **level-2**} {**route-map** *route-map* | **all**}
7. (Optional) **show isis** [**vrf** *vrf-name*] {**ip** | **ipv6**} **route** *ip-prefix* [*detail* | **longer-prefixes** [**summary** | **detail**]]
8. (Optional) **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	Use the **configure terminal** command to enter global configuration mode.
	Example:
	`switch# configure terminal`
	`switch(config)#`	

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* | {**eigrp** | **isis** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static** | **direct**} **route-map** *map-name* **Example:** `switch(config-router-af)# redistribute eigrp 201 route-map ISISmap`	Redistributes routes from other protocols into IS-IS.
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-router-af)# show isis ip route`	Shows the IS-IS routes.
Step 8	(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 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.

SUMMARY STEPS

1. Use the **configure terminal** command to enter global configuration mode.
2. **router isis** *instance-tag*
3. **redistribute** {**bgp** *id* | **direct** | **eigrp***id* | **isis** *id* | **ospf** *id* | **rip** *id* | **static**} **route-map** *map-name*
4. **redistribute maximum-prefix** *max* [*threshold*] [**warning-only** | **withdraw** [*num-retries timeout*]]
5. (Optional) **show running-config isis**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	Use the **configure terminal** command to enter global configuration mode.
	Example:
	`switch# configure terminal`
	`switch(config)#`	

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:**	Specifies a maximum number of prefixes that IS-IS distributes. The range is from 1 to 65535. You can optionally specify the following:

	Command or Action	Purpose
	`switch(config-router)# redistribute maximum-prefix 1000 75 warning-only`	• *threshold*—Percent 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 isis Enterprise
switch(config-router)# redistribute bgp route-map FilterExternalBGP
switch(config-router)# redistribute maximum-prefix 1000 75
```

Advertise Only Passive Interface Prefixes

You can specify that only prefixes belonging to passive interfaces are advertised in the system link-state packets (LSPs).

Procedure

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 200
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 **[no] advertise passive-only {level-1 | level-2}**

Example:

```
switch(config-router-af)# advertise passive-only level-1
switch(config-router-af)#
```

Enables the advertisement of only those prefixes that belong to passive interfaces.

Example

This example shows how to enable only the advertising of prefixes belonging to passive interfaces:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>interface ethernet 1/2</userinput>
switch(config-if)# <userinput>address-family ipv4 unicast</userinput>
switch(config-router-af)# <userinput>advertise passive-only level-1</userinput>
```

Suppress Prefixes on an Interface

You can allow an IS-IS interface to participate in forming adjacencies without advertising connected prefixes in the system link-state packets (LSPs).

Procedure

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] isis suppress**

Example:

```
switch(config-if)# isis suppress
switch(config-if)#
```

Disables the advertisement of connected prefixes on the interface.

Example

This example shows how to suppress the advertising of an interface's connected prefixes in the system link-state packets (LSPs):

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>interface ethernet 1/2</userinput>
switch(config-if)# <userinput>isis suppress</userinput>
```

Disable 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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

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 unicast
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.

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 **no adjacency-check**

Example:

```
switch(config-router-af)# no
adjacency-check
```

Disables strict adjacency mode for the IPv6 address family.

Step 8 (Optional) **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.

Configure 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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

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
```

Enables a graceful restart and the graceful restart helper functionality. Enabled by default.

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
```

Copies the running configuration to the startup configuration.

Example

This example shows how to enable a graceful restart:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router isis Enterprise</userinput>
switch(config-router)# <userinput>graceful-restart</userinput>
switch(config-router)# <userinput>copy running-config startup-config</userinput>
```

Configure 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.

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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

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 (Optional) **vrf***vrf-name*

Example:

```
switch(config-router)# vrf
RemoteOfficeVRF
switch(config-router-vrf)#
```

Enters router 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 **exit**

Example:

```
switch(config-router)# exit
switch(config)#
```

Exits router configuration mode.

Step 9 **interface ethernet***slot/port*

Example:

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

Enters interface configuration mode.

Step 10 **vrf member***vrf-name*

Example:

```
switch(config-if)# vrf member
RemoteOfficeVRF
```

Adds this interface to a VRF.

Step 11 {**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 complete this step after you assign this interface to a VRF.

Step 12 {**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 13 (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 14 (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# <userinput>configure terminal</userinput>
switch(config)# <userinput>vrf context NewVRF</userinput>
switch(config-vrf)# <userinput>exit</userinput>
switch(config)# <userinput>router isis Enterprise</userinput>
switch(config-router)# <userinput>vrf NewVRF</userinput>
switch(config-router-vrf)# <userinput>net 47.0004.004d.0001.0001.0c11.1111.00</userinput>
switch(config-router-vrf)# <userinput>exit</userinput>
switch(config-router)# <userinput>exit</userinput>
switch(config)# <userinput>interface ethernet 1/2</userinput>
switch(config-if)# <userinput>vrf member NewVRF</userinput>
switch(config-if)# <userinput>ip address 192.0.2.1/16</userinput>
switch(config-if)# <userinput>ip router isis Enterprise</userinput>
switch(config-if)# <userinput>copy running-config startup-config</userinput>
```

Tune IS-IS

You can tune IS-IS to match your network requirements.

You can use the following optional commands to tune IS-IS:

Procedure

Step 1 (Optional) **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.

Step 2 (Optional) **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.

Step 3 (Optional) **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.

Step 4 (Optional) **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.

Step 5 (Optional) **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.

Step 6 (Optional) **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.

Step 7 (Optional) **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.

Step 8 (Optional) **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.
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.

Command	Purpose
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.

Monitor 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 ip 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

Command	Purpose
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:

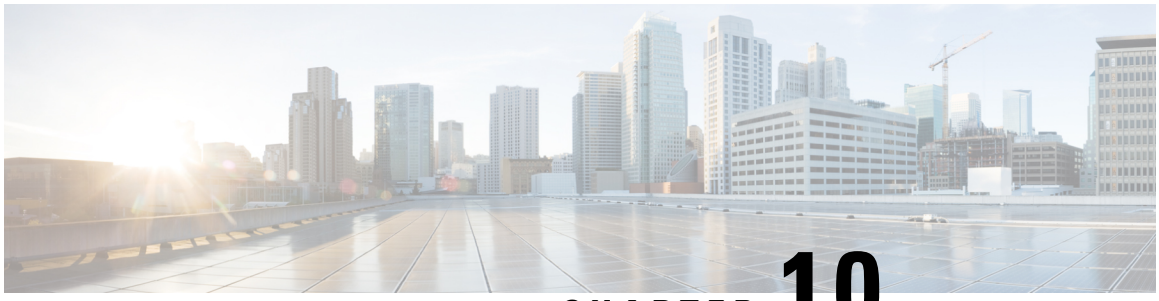
```
router isis Enterprise
        is-type level-1
        net 49.0001.0000.0000.0003.00
        graceful-restart
        address-family ipv4 unicast
        default-information originate

        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, on page 523 for more information on route maps.

Related Topics



CHAPTER 10

BGP

- BGP, on page 287
- Prerequisites, on page 299
- Guidelines and limitations for basic and advanced BGP, on page 299
- Default Settings, on page 305
- Configure Basic BGP, on page 305
- Configure advance BGP, on page 324

BGP

Border Gateway Protocol (BGP) is an inter-domain routing protocol that provides loop-free routing between organizations or autonomous systems.

Cisco NX-OS supports BGP version 4. This version 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 devices called BGP peers. When connecting to an external organization, a router creates external BGP (eBGP) peering sessions. When connecting to a BGP peer within the same organization to exchange routing information, a router creates internal BGP (iBGP) peering sessions.

BGP uses a path-vector routing algorithm to exchange routing information among BGP-enabled networking devices or BGP speakers. Based on this information, each BGP speaker determines a path to reach a destination while detecting and avoiding paths with routing loops. The routing information includes the prefix for a destination, the path of autonomous systems to the destination, and other path attributes.

By default, BGP selects a single path as the best path to a destination host or network. Each path includes well-known mandatory attributes, well-known discretionary attributes, and optional transitive attributes for BGP best-path analysis. You can influence BGP path selection by altering some of these attributes by configuring BGP policies. For more information, see Route Policies and Resetting BGP Sessions. BGP also supports load balancing or equal-cost multipath (ECMP). For more information, see Load Sharing and Multipath.

Beginning with Cisco NX-OS Release 10.5(1)F, Configuring Basic BGP and Configuring Advanced BGP chapters are merged to create Configuring BGP chapter.

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 autonomous system (AS) numbers in plain-text notation or as.dot notation and 4-byte AS numbers in plain-text notation.

When BGP is configured with a 4-byte AS number, the **route-target auto** VXLAN command cannot be used because the AS number along with the VNI (which is already a 3-byte value) is used to generate the route target. For more information, see the Cisco Nexus 9000 Series NX-OS VXLAN Configuration Guide.

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 table.

Table 22: 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	220	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 peer is a BGP speaker that has an active TCP connection to another BGP speaker.

A BGP speaker does not discover another BGP speaker automatically. You must configure the relationships between BGP speakers.

BGP sessions

BGP sessions are TCP connections established between Border Gateway Protocol (BGP) peers that enable the exchange of routing information. BGP uses TCP port 179 to create these sessions, which serve as the communication channel for exchanging routing updates and maintaining network topology awareness.

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 these 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 and interface peers

Dynamic AS numbers are a feature in Cisco NX-OS BGP configurations that allow you to specify a range or list of Autonomous System (AS) numbers for establishing BGP sessions with prefix peers or interface peers. This means BGP will accept sessions only from peers whose AS numbers match those in the configured list or range.

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.

Beginning with Cisco NX-OS Release 9.3(6), support for dynamic AS numbers is extended to interface peers in addition to prefix peers. See Configure BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families, on page 347.

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. For more information, see *Configuring Advanced BGP*.

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, 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.

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, then Cisco NX-OS prefers the first loopback interface over any other interface type.
- If you have not configured a loopback interface, 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 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.

BGP path selection

BGP best-path algorithm is a process used by Cisco NX-OS to select the optimal path for routing a given network prefix when multiple valid paths are available. For information on configuring additional BGP paths, see Configuring Advance BGP.

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 these steps:

1. Compares two paths to determine which is better. See the Step 1 Comparing Pairs of Paths section).
2. Explores all paths and determines in which order to compare the paths to select the overall best path. See the Step 2 Determining the Order of Comparisons section.
3. Determines whether the old and new best paths differ enough so that the new best path should be used. See the Step 3 Determining the Best-Path Change Suppressionsection.



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.

Note VXLAN deployments use a BGP path selection process that differs from the normal selection of local over remote paths. For the EVPN address family, BGP compares the sequence number in the MAC Mobility attribute (if present) and selects the path with the higher sequence number. If both paths being compared have the attribute and the sequence numbers are the same, BGP prefers the path that is learned from the remote peer over a locally originated path. For more information, see the Cisco Nexus 9000 Series NX-OS VXLAN Configuration Guide.

BGP path selection - comparing pairs of paths

This first step in the BGP best-path algorithm compares two paths to determine which path is better. These following sequences describe 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, on page 327 section for more information.

6. Cisco NX-OS chooses the path with the lower origin. Interior Gateway Protocol (IGP) is considered lower than EGP.
7. Cisco NX-OS chooses the path with the lower multiexit discriminator (MED).

   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 Tune the best-path algorithm, on page 334 section for more information. Otherwise, Cisco NX-OS performs a MED comparison that depends on the AS-path attributes of the two paths being compared.

   You can configure Cisco NX-OS to always perform the best-path algorithm MED comparison, regardless of the peer autonomous system in the paths. Otherwise, Cisco NX-OS performs a MED comparison that depends on the AS-path attributes of the two paths being compared these ways:

   a. If a path has no AS-path or the AS-path starts with an AS_SET, 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, 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 that are followed by an AS_SEQUENCE, 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, 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 Tune the best-path algorithm, on page 334 for more information.

e. If the non-deterministic MED comparison feature is enabled, the best-path algorithm uses the Cisco IOS style of MED comparison.

8. If one path is from an internal peer and the other path is from an external peer, Cisco NX-OS chooses the path from the external peer.
9. If the paths have different IGP metrics to their next-hop addresses, 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, you can configure the best-path algorithm to enforce comparison of the router IDs when both paths are eBGP by configuring “compare router-id”. In all other cases, the router-id comparison is done by default.

    See the Tune the best-path algorithm, on page 334 for more information. If the path includes an originator attribute, 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.

Note 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, so you could 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.

Note Paths that are equal after Step 9 can be used for multipath if you configure multipath. See the Load sharing and multipath, on page 329 section for more information.

BGP path selection - determining the order of comparisons

In the second step of the BGP best-path algorithm implementation, Cisco NX-OS uses these steps to 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 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, 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.

BGP path selection - determining the best-path change suppression

The next part of the implementation is to determine whether Cisco NX-OS uses the new best path or suppresses 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 Tune the best-path algorithm, on page 334 section for more information. If you configure this feature, the new best path is always preferred to the existing one.

BGP and the unicast RIB

BGP communicates with the unicast routing information base (unicast RIB) to store IPv4 and IPv6 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 communicates with the IPv6 unicast RIB to perform these operations for IPv6 routes.

BGP prefix independent convergence

The BGP prefix independent convergence (PIC) edge feature achieves faster convergence in the forwarding plane for BGP IP routes to a BGP backup path when there is a link failure.

The BGP PIC edge feature improves BGP convergence after a network failure. This convergence applies to edge failures in an IP network. This feature creates and stores a backup path in the routing information base (RIB) and forwarding information base (FIB) so that when the primary path fails, the backup path can immediately take over, enabling fast failover in the forwarding plane. BGP PIC edge supports only IPv4 address families.

When BGP PIC edge is configured, BGP calculates a second-best path (the backup path) along with the primary best path. BGP installs both best and backup paths for the prefixes with PIC support into the BGP RIB. BGP also downloads the backup path along with the remote next hop through APIs to the URIB, which then updates the FIB with the next hop marked as a backup. The backup path provides a fast reroute mechanism to counter a singular network failure.

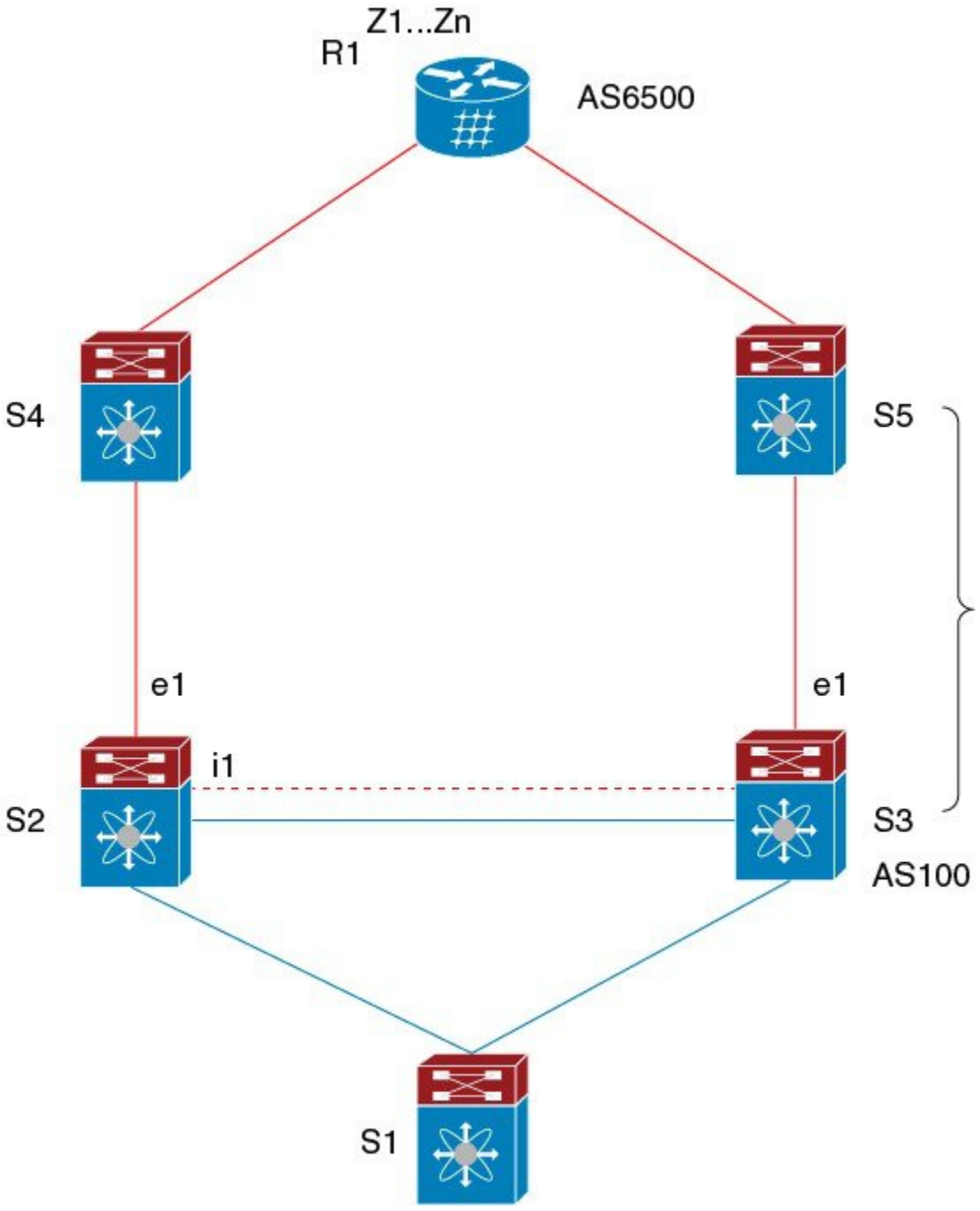
This feature detects both local interface failures and remote interface or link failures and triggers the use of the backup path.

BGP PIC edge supports both unipath and multipath.

BGP PIC edge unipath

Figure GBP PIC Edge Unipath shows a BGP PIC edge unipath topology.

Figure 27: BGP PIC Edge Unipath



In this figure:

- eBGP sessions are between S2-S4 and S3-S5.
- The iBGP session is between S2-S3.

- Traffic from S1 uses S2 and uses the e1 interface to reach prefixes Z1...Zn.
- S2 has two paths to reach Z1…Zn:
  - A primary path through S4
  - A backup path through S5

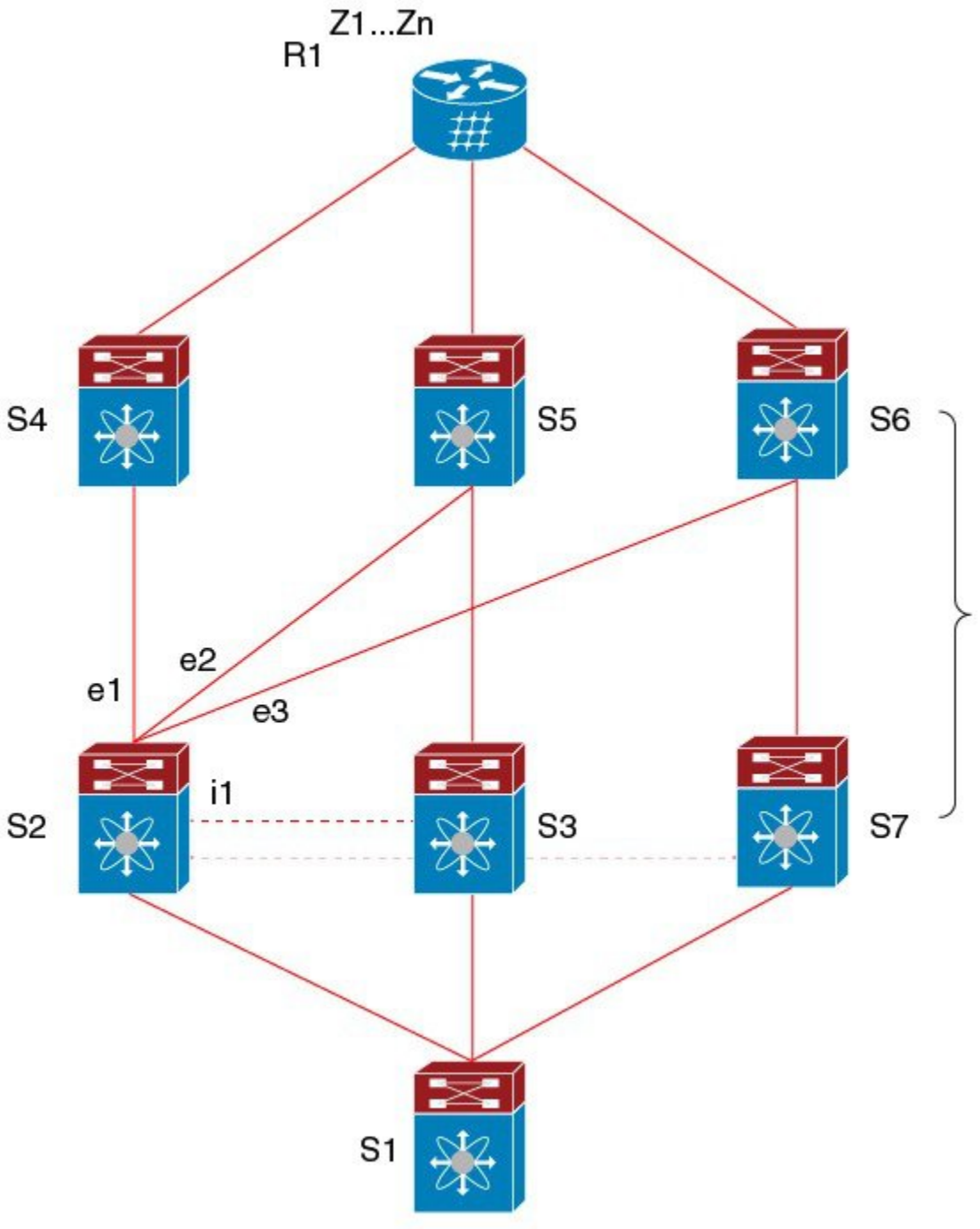
In this example, S3 advertises to S2 the prefixes Z1…Zn to reach (with itself as the next hop). With BGP PIC edge enabled, BGP on S2 installs both the best path (through S4) and the backup path (through S3 or S5) toward the AS6500 into the RIB. Then the RIB downloads both routes to the FIB.

If the S2-S4 link goes down, the FIB on S2 detects the link failure. It automatically switches from the primary path to the backup path and points to the new next hop S3. Traffic is quickly rerouted due to the local fast re-convergence in the FIB. After learning of the link failure event, BGP on S2 recomputes the best path (which is the previous backup path), removes next hop S4 from the RIB, and reinstalls S3 as the primary next hop into the RIB. BGP also computes a new backup path, if any, and notifies the RIB. With the support of the BGP PIC edge feature, the FIB can switch to the available backup route instantly upon detection of a link failure on the primary route without waiting for BGP to select the new best path and converge to achieve a fast reroute.

BGP PIC edge with multipath

The following figure shows a BGP PIC edge multipath topology.

Figure 28: BGP PIC Edge Multipaths



In this topology, there are six paths for a given prefix:

- eBGP paths: e1, e2, e3
- iBGP paths: i1, i2, i3

The order of preference is e1 > e2 > e3 > i1 > i2 > i3.

The potential multipath situations are:

- No multipaths configured:
  - bestpath = e1
  - multipath-set = []
  - backup path = e2
  - PIC behavior: When e1 fails, e2 is activated.
- Two-way eBGP multipaths configured:
  - bestpath = e1
  - multipath-set = [e1, e2]
  - backup path = e3
  - PIC behavior: Active multipaths are mutually backed up. When all multipaths fail, e3 is activated.
- Three-way eBGP multipaths configured:
  - bestpath = e1
  - multipath-set = [e1, e2, e3]
  - backup path = i1
  - PIC behavior: Active multipaths are mutually backed up. When all multipaths fail, i1 is activated.
- Four-way eBGP multipaths configured:
  - – bestpath = e1
  - – multipath-set = [e1, e2, e3, i1]
  - – backup path = i2
  - – PIC behavior: Active multipaths are mutually backed up. When all multipaths fail, i2 is activated.

When the Equal Cost Multipath Protocol (ECMP) is enabled, none of the multipaths can be selected as the backup path.

For multipaths with the backup path scenario, faster convergence is not expected with simultaneous failure of all active multipaths.

BGP PIC core

BGP Prefix Independent Convergence (PIC) in Core is a BGP optimization technique that

- improves BGP convergence speed after a network failure,
- reduces the time and resources needed to update forwarding information for multiple prefixes, and
- enables immediate leveraging of IGP convergence by hierarchical FIB programming.

When a link fails on Provider Edge (PE), the Routing Information Base (RIB) updates the Forwarding Information Base (FIB) with new next hop. FIB must update all BGP prefixes that point to the failed next hop and point to the new one. This can be time and resource consuming. With BGP PIC Core enabled, the prefix is programmed in the FIB in a hierarchical way. All prefixes point to the ECMP group instead of the recursive next hop. When the same failure happens, the FIB only needs to update the ECMP group to point to the new next hop without updating prefixes. This gives BGP immediate leveraging of IGP convergence.

BGP PIC feature support matrix

Table 23: BGP PIC Feature Support Matrix

BGP PIC	IPv4 Unicast	IPv6 Unicast
Edge unipath	Yes	No
Edge with multipath (multiple active ECMPs, only one backup)	Yes	No
Core	Yes	Yes

Note The PIC Core (**system pic-core** command) and PIC Edge must be used exclusively in a Layer 3 environment and are not compatible with VXLAN environment.

BGP virtualization

BGP supports virtual routing and forwarding (VRF) instances.

Prerequisites

BGP has these prerequisites:

- You must enable BGP.
- 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 basic and advanced BGP

BGP has these configuration guidelines and limitations:

- With sufficient scale (such as - hundreds of peers and thousands of routes per peer), the Graceful Restart mechanism may fail because the default 5 minute stale-path timer might not be enough for BGP convergence to complete before the timer expires. Use the following command to verify the actual time taken for the convergence process:

```
switch# show bgp vrf all all neighbors | in First|RIB
  Last End-of-RIB received 0.022810 after session start
  Last End-of-RIB sent 00:08:36 after session start
  First convergence 00:08:36 after session start with 398002 routes sent
```

- Do not leave persistent BGP debugging (debug bgp file) or detailed BGP event-history logging enabled during high-scale operation. Disable this logging before disruptive events, such as a BGP restart or an ASCII reload. This logging can consume significant CPU resources and severely delay BGP convergence and traffic recovery.

  Sample configuration:

```
router bgp 65001
  debug bgp file
  event-history periodic size large
  event-history detail size large
```

- Beginning with Cisco NX-OS 9.3(5), a packet with a TTL value of 1 to a vPC peer is hardware forwarded.
- For large routing tables (250 K or above) when using the SNMP bulkwalk with record option (-Cr), do not use more than 10 records to avoid SNMP performance degradation.
- There are three scenarios in which the command behavior has changed beginning with Cisco NX-OS Release 9.3(5):

  - ```
    Router bgp 1
         Template peer abc
             Ttl-security hops 30
         Neighbor 1.2.3.4
             Inherit peer abc
    ```

    If you later enter the **ebgp-multihop 20** command, the configuration is blocked due to the presence of **ttl-security hops 30** command. Beginning with the Cisco NX-OS Release 9.3(5), the configuration is no longer blocked. However, the **ttl-security hops** command has priority and would be the enabled functionality.

  - ```
    Router bgp 1
         Template peer abc
             Ebgp-multihops 20
         Neighbor 1.2.3.4
             Inherit peer abc
    ```

    If you later enter the **ttl-security hops 30** command, the configuration is blocked due to the presence of **ebgp-multihop 20** command. Beginning with Cisco NX-OS Release 9.3(5), the configuration is no longer blocked. However again, the **ttl-security hops** command has priority and would be the enabled functionality.

  - ```
    Router bgp 1
         Template peer abc
             Remote-as 1
         Neighbor 1.2.3.4
             Inherit peer abc
    ```

    If you later enter the **ttl-security hops 30** or **ebgp-multihop 20** commands, they are blocked. Beginning with Cisco NX-OS Release 9.3(5), the configuration is not blocked. However, their
    ```

functionalities are turned off as the **remote-as** command has priority which makes the peer an iBGP peer.

- If you are writing JSON payload, use the standard JSON syntax defined on RFC 8259: The JavaScript Object Notation (JSON) Data Interchange Format.
- Prefix peering operates only in passive TCP mode. It accepts incoming connections from remote peers if the peer address falls within the prefix.
- Beginning with Cisco NX-OS 9.3(5), a packet with a TTL value of 1 to a vPC peer is hardware that is forwarded.
- Configuring the **advertise-maps** command multiple times is not supported.
- Names in the prefix-list are case-insensitive. We recommend using unique names. Do not use the same name by modifying uppercase and lowercase characters. For example, CTCPrimaryNetworks and CtcPrimaryNetworks are not two different entries.
- The dynamic AS number prefix peer configuration overrides the individual AS number configuration that is 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.
- When you configure BGP confederation peers using configure staging, the order of the AS numbers is not preserved after you commit. The candidate difference preview shows the correct order as entered, but after you commit, the running config displays the peers in a different order. This behavior does not occur in regular configuration mode, which correctly preserves the order as entered.
- BGP sessions that are 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 that are received and system resources used.
- Beginning with Cisco NX-OS Release 10.6(2)F, a new option discard is introduced in maximum-prefix configuration for BGP. You can configure this option to discard excess prefixes received from the neighbor once the threshold is reached.
  - This feature does not support ISSD.
  - The existing NX-OS BGP maximum-prefix implementation is post-inbound policy. Thus, if soft-reconfig is enabled, BGP will still store the received routes. However, the discarded routes will not be installed in BRIB. Currently there is no option to apply this configuration at a pre-inbound stage.
  - BGP cannot guarantee hitless graceful restart in the presence of discarded prefixes.
- Configure the update source to establish a session with BGP/eBGP multihop sessions.
- Specify a BGP policy if you configure redistribution.
- Define the BGP router ID within a VRF.
- For IPv6 neighbors, Cisco recommends that you configure a router ID per VRF. If a VRF does not have any IPv4 interfaces, the IPv6 BGP neighbor will not come up because its router ID must be an IPv4

address. The numerically lowest loopback IPv4 address is elected to be the router ID. If a loopback address does not exist, the lowest IP address from the VRF interfaces is elected. If that does not exist, the BGP neighbor relationship is not established.

- If you decrease the keepalive and hold timer values, you might experience BGP session flaps.
- You can configure a minimum route advertisement interval (MRAI) between the sending of BGP routing updates by using the **advertisement-interval** command.
- Although the **show ip bgp** commands are available for verifying the BGP configuration, Cisco recommends that you use the **show bgp** commands instead.
- When you redistribute BGP to IGP, iBGP is redistributed as well. To override this behavior, you must insert an extra deny statement into the route map.
- Edge Services Gateway (ESG) route-inject into BGP is assigned weight 0.
- To enable BFD for iBGP single-hop peers, you must configure the **update-source** option on the physical interface.
- Beginning with Cisco NX-OS Release 9.3(3), BFD for BGP is supported for BGP IPv4 and IPv6 prefix peers.
- The following guidelines and limitations apply to the **remove-private-as** command:
  - It applies only to eBGP peers.
  - It applies only to routers in a public AS only. The workaround to this restriction would be to apply the **neighbor local-as** command on a per-neighbor basis, with the local AS number being a public AS number.
  - It can be configured only in neighbor configuration mode and not in neighbor-address-family mode.
  - If the AS-path includes both private and public AS numbers, the private AS numbers are not removed.
  - If the AS-path contains the AS number of the eBGP neighbor, the private AS numbers are not removed.
  - Private AS numbers are removed only if all AS numbers in that AS-path belong to a private AS number range. Private AS numbers are not removed if a peer's AS number or a non-private AS number is found in the AS-path segment.
- In case the AS-Path is missing from the matching route for the next-hop in the BGP table or there is no matching route for the nexthop in the BGP table then the AS number is not printed in the traceroute output along with the next hop.
- If you use the **aggregate-address** command to configure aggregate addresses and the **suppress-fib-pending** command to suppress BGP routes, lossless traffic for aggregates cannot be ensured on BGP or system triggers.
- When you enable FIB suppression on the switch and route programming fails in the hardware, BGP advertises routes that are not programmed locally in the hardware.
- If you disable a command in the neighbor, template peer, template peer-session, or template peer-policy configuration mode (and the **inherit peer** or **inherit peer-session** command is present), you must use the **default** keyword to return the command to its default state. For example, to disable the **update-source loopback 0** command from the running configuration, you must enter the **default update-source loopback 0** command.

- When next-hop-self is configured for route-reflector clients, the route reflector advertises routes to its clients with itself as the next hop.
- The following guidelines and limitations apply to weighted ECMP:
  - Weighted ECMP is supported only for the IPv4 address family.
  - BGP uses the Link Bandwidth EXTCOMM defined in the draft-ietf-idr-link-bandwidth-06.txt to implement the weighted ECMP feature.
  - BGP accepts the Link Bandwidth EXTCOMM from both iBGP and eBGP peers.
- The following guidelines and limitations apply to BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families:
  - This feature does not support having the same link-local address configured across multiple interfaces.
  - While configuring BGP interface peering using IP6 Link-Local static IPv6 address, ensure the subnet is matching with the peer for BGP to work.
  - This feature is not supported on logical interfaces (loopback). Only Ethernet interfaces, port-channel interfaces, subinterfaces, and breakout interfaces are supported.
  - Beginning with Cisco NX-OS Release 9.3(6), VLAN interfaces are supported.
  - This feature is supported only for IPv6-enabled interfaces with link-local addresses.
  - This feature is not supported when the configured prefix peer and interface have the same remote peer.
  - The following commands are not supported in neighbor interface configuration mode:
    - **disable-connected-check**
    - **maximum-peers**
    - **update-source**
    - **ebgp-multihop**
  - BFD multihop and the following commands are not supported for BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families:
    - **bfd-multihop**
    - **bfd multihop interval**
    - **bfd multihop authentication**
  - BGP requires faster convergence time for route advertisements. To speed up detection of the Route Advertisement (RA) link-level protocol, enter the following commands on each IPv6-enabled interface that is using BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families:

```
interface Ethernet
                        port/slot

ipv6 nd ra-interval 4 min 3
ipv6 nd ra-lifetime 10
```

- Beginning with Cisco NX-OS Release 9.3(1)F, path type is introduced in the **rn** key to distinguish among the multiple entries with different path type while using REST API to see BGP routing table for IPv4 unicast routes. In the earlier releases, the output is displayed as single entry.
- Route-map deletion feature adds a mechanism to block the deletion of entire route-map that is associated with the BGP. With the route-map deletion blocked, the modifications to the route-map statement are still allowed.
- If there are more than one sequence in the route-map, user can still delete any route map sequence until there is at least one sequence available.
- Users can have the forward reference case for route-map from client. However, once route-map is created and associated, the deletion of route-map is blocked.
- Blocking deletion functionality is configurable dynamically using the knob.
- It is allowed to delete the BGP association to the route-map and deletion of route-map itself in a single transaction payload.
- It is allowed to add the BGP association to the route-map and an error must be thrown for deletion of route-map.
- The following is the list of the dual stage related behaviors:
  - If knob and deletion occur together, dual stage has to verify and throw an error without commit.
  - If knob already exists and route-map deletion occurs in dual stage, it must throw an error.
  - If route-map and CLI knob is single commit with different order, it must throw an error.
  - If knob is not enabled and route-map deletion occurs in dual stage, it has to execute successfully.
  - In a single verify, if "cli knob is disabled AND route-map deletion" is executed, the route-map deletion is allowed.
- If the route-map used by BGP template is not inherited by any of the BGP neighbors, the enitre route-map deletion will still be blocked.
- Default-originate route-map only supports prefix-based matching. Using any BGP attribute-based match will be silently ignored, effectively making the route-map a permit-all. The default route will always be advertised in such cases.
- Add-paths selection route-map only supports matching prefix lists. There is no support for attribute matching.
- There are few commands under vrf context that are owned by BGP, but are not part of bgpInst.
- As the VPN address family (L3VPN and EVPN) is not supported, the routes received from confederate peers are not advertised in the VPN address family.
- Cloudscale IPv6 link-local BGP support requires carving > 512 ing-sup TCAM region (this requires a reload to take effect).
- As the VPN address family (L3VPN and EVPN) is not supported, the routes received from confederate peers are not advertised in the VPN address family.
- Beginning with Cisco NX-OS Release 10.3(1)F, BGP is supported on the Cisco Nexus 9808 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, BGP is supported on the Cisco Nexus 9804 switches.

- Beginning with Cisco NX-OS Release 10.3(1)F, VXLAN EVPN is supported only as transit on Cisco Nexus 9808 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, VXLAN EVPN is supported only as transit on Cisco Nexus 9804 switches.
- Encryption decrypt type6 is not supported for BGP passwords and keychain.
- Beginning with Cisco NX-OS Release 10.4(1)F, BGP is supported on Cisco Nexus X98900CD-A and X9836DM-A line cards with 9808 and 9804 switches.
- Beginning with Cisco NX-OS Release 10.6(1)F, BGP is supported on Cisco Nexus N9336C-SE1 platform switches.
- Beginning with Cisco NX-OS Release 10.6(1)F, ECMP is supported on Cisco Nexus N9336C-SE1 platform switches.
- Beginning with Cisco NX-OS Release 10.6(2)F, BGP, ECMP, 8192 ECMP groups, and 512-way ECMP are supported on Cisco Nexus N9324C-SE1U and N9348Y2C6D-SE1U platform switches.

Default Settings

Table 24: Default BGP Parameters

Parameters	Default
BGP feature	Disabled
Keep alive interval	60 seconds
Hold timer	180 seconds
BGP PIC edge	Disabled
Auto-summary	Always disabled
Synchronization	Always disabled

Configure Basic BGP

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 Advance BGP*.

This example shows how to enter router configuration mode:

```
switch# configuration
switch(config)# router bgp 64496
switch(config-router)#
```

BGP supports 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.

The following 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)#
```

The following 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.

The following 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)#
```

The following 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 the introduction of 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 or 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)#
```

Configure Basic BGP

To configure a basic BGP, you must 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.



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.

Enable BGP

You must enable BGP before you can configure BGP.

Procedure

Step 1 **configure terminal**

Example:

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

Enters configuration mode.

Step 2 **[no] feature bgp**

Example:

```
switch(config)# feature bgp
```

Enables BGP.

Use the **no** form of this command to disable this 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.

Create a BGP Instance

You can create a BGP instance and assign a router ID to the BGP instance. For more information, see *BGP Router Identifier* section.

Before you begin

- You must enable BGP (see the Enabling BGP section).
- BGP must be able to obtain a router ID (for example, a configured loopback address).

Procedure

Step 1 **configure terminal**

Example:

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

Enters configuration mode.

Step 2 [**no**] **router bgp** {*autonomous-system-number* | *auto*}

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.

Auto option generates 4-Byte Private Autonomous System Number automatically based on system MAC address.

Use the **no** option with this command to remove the BGP process and the associated configuration.

Step 3 **router-id** {*ip-address* | *auto*}

Example:

```
switch(config-router)# router-id 192.0.2.255
```

(Optional) Configures the BGP router ID. This IP address identifies this BGP speaker.

"auto" option will enable the BGP router ID based on system MAC address.

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 IPv4 or IPv6 address family.

Step 5 (Optional) **network** {*ip-address/length* | *ip-address* **mask** *mask*} [**route-map** *map-name*]

Example:

```
switch(config-router-af)# network 10.10.10.0/24
```

Example:

```
switch(config-router-af)# network 10.10.10.0 mask 255.255.255.0
```

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 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

This example shows how to enable BGP with the IPv4 unicast address family and manually add one network to advertise:

```
switch# <userinput>configure terminal</userinput>
            switch(config)# <userinput>router bgp 64496</userinput>
            switch(config-router)# <userinput>address-family ipv4 unicast</userinput>
            switch(config-router-af)# <userinput>network 192.0.2.0</userinput>
            switch(config-router-af)# <userinput>copy running-config
startup-config</userinput>
```

Restart 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:

Procedure

restart bgp*instance-tag*

Example:

```
switch(config)# restart bgp 201
```

Restarts the BGP instance and resets or reestablishes all peering sessions.

Shut Down BGP

You can shut down the BGP protocol and gracefully disable BGP while retaining the configuration.

To shut down BGP, use the following command in router configuration mode:

Procedure

shutdown

Example:

```
switch(config-router)# shutdown
```

Restarts the BGP instance and resets or reestablishes all peering sessions.

Configure 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

- You must enable BGP (see the Enabling BGP section).

Procedure

Step 1 **configure terminal**

Example:

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

Enters 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* | *ipv6-address*} **remote-as** {*as-number* | *external* | *internal*}

Example:

```
switch(config-router)# neighbor
209.165.201.1 remote-as 64497
switch(config-router)# neighbor
```

Configures the IPv4 or IPv6 address and AS number for a remote BGP peer. *The ip-address* format is x.x.x.x. The *ipv6-address* format is A:B::C:D.

The external and internal options allow eBGP and iBGP sessions to be established without manually providing remote-as values.

Step 4 **remote-as** {*as-number* | *external* | *internal*}

Example:

```
switch(config-router-neighbor)# remote-as 64497
```

Configures the AS number for a remote external BGP peer.

The external and internal options allow eBGP and iBGP sessions to be established without manually providing remote-as values.

Step 5 (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.

Step 6 (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 value of keepalive time is 60 seconds and hold time is 180 seconds.

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 7 (Optional) **shutdown**

Administratively shuts down this BGP neighbor. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

Example:

```
switch(config-router-neighbor)# shutdown
```

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 unicast IPv4 or IPv6 address family.

Step 9 (Optional) **weight** *value*

Example:

```
switch(config-router-neighbor-af)#
weight 100
```

Sets the default weight for routes from this neighbor. The range is from 0 to 65535.

All routes learned from this neighbor have the assigned weight initially. The route with the highest weight is chosen as the preferred route when multiple routes are available to a particular network. The weights assigned with the **set weight route-map** command override the weights assigned with this command.

If you specify a BGP peer policy template, all the members of the template inherit the characteristics configured with this command.

Step 10 (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 11 (Optional) **copy running-config startup-config**

Example:

```
switch(config-router-neighbor-af)# copy
running-config startup-config
```

Saves this configuration change.

Example

The following example shows how to configure a BGP peer:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router bgp 64496</userinput>
switch(config-router)# <userinput>neighbor 192.0.2.1 remote-as 64497</userinput>
switch(config-router-neighbor)# <userinput>description Peer Router B</userinput>
switch(config-router-neighbor)# <userinput>address-family ipv4 unicast</userinput>
switch(config-router-neighbor)# <userinput>weight 100</userinput>
switch(config-router-neighbor-af)# <userinput>copy running-config startup-config</userinput>
```

Configure 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

- You must enable BGP (see the Enabling BGP section).

Procedure

Step 1 **configure terminal**

Example:

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

Enters 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 or IPv6 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 *prefix* format for IPv6 is A:B::C:D/length. The length range is from 1 to 128.

The *map-name* can be any case-sensitive, alphanumeric string up to 63 characters.

Step 4 **neighbor-as** *as-number*

Example:

```
switch(config-router-neighbor)# remote-as 64497
```

Configures the AS number for a remote BGP peer.

Step 5 (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 6 (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 dynamic AS numbers for a prefix peer:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>route-map BGPPeers</userinput>
switch(config-route-map)# <userinput>match as-number 64496, 64501-64510</userinput>
switch(config-route-map)# <userinput>match as-number as-path-list List1, List2</userinput>
switch(config-route-map)# <userinput>exit</userinput>
switch(config)# <userinput>router bgp 64496</userinput>
switch(config-router)# <userinput>neighbor 192.0.2.0/8 remote-as route-map
BGPPeers</userinput>
switch(config-router-neighbor)# <userinput>description Peer Router B</userinput>
switch(config-router-neighbor)# <userinput>address-family ipv4 unicast</userinput>
switch(config-router-af)# <userinput>end</userinput>
switch# <userinput>copy running-config startup-config</userinput>
```

See Configuring Route Policy Manager, on page 523 for information on route maps.

Configure BGP PIC Edge

Follow these steps to configure BGP PIC edge.

Note The BGP PIC edge feature supports only IPv4 address families.

Before you begin

You must enable BGP (see the Enabling BGP section).

Procedure

Step 1 **configure terminal**

Example:

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

Enters 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 **address-family ipv4 unicast**

Example:

```
switch(config-router)# address-family ipv4 unicast
                        switch(config-router-af)#
```

Enters address family configuration mode for the IPv4 address family.

Step 4 **[no] additional-paths install backup**

Example:

```
switch(config-router-af)#
                        [no] additional-paths install backup
```

Enables BGP to install the backup path to the routing table.

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

Example:

```
switch(config-router-af)# end
switch# copy running-config startup-config
```

Saves this configuration change.

Example

This example shows how to configure the device to support BGP PIC edge in an IPv4 network:

```
interface Ethernet2/2
ip address 1.1.1.5/24
no shutdown

interface Ethernet2/3
ip address 2.2.2.5/24
no shutdown

router bgp 100
address-family ipv4 unicast
additional-paths install backup
neighbor 2.2.2.6
remote-as 100
address-family ipv4 unicast
```

If BGP receives the same prefix (for example, 99.0.0.0/24) from the two neighbors 1.1.1.6 and 2.2.2.6, both paths are installed in the URIB, one as the primary path and the other as the backup path.

BGP output:

```
switch(config)# show ip bgp 99.0.0.0/24
BGP routing table information for VRF default, address family IPv4 Unicast
BGP routing table entry
for 99.0.0.0/24, version 4
Paths: (2 available, best #2)
Flags: (0x00001a) on xmit-list, is in urib, is best urib route

Path type: internal, path is valid, not best reason: Internal path, backup
path AS-Path: 200 , path
sourced external to AS
2.2.2.6 (metric 0) from 2.2.2.6 (2.2.2.6)
Origin IGP, MED not set, localpref 100, weight 0

Advertised path-id 1
Path type: external, path is valid, is best path AS-Path: 200 , path sourced
external to AS
1.1.1.6 (metric 0) from 1.1.1.6 (99.0.0.1)
Origin IGP, MED not set, localpref 100, weight 0

Path-id 1 advertised to peers: 2.2.2.6
```

URIB output:

```
switch(config)# show ip route 99.0.0.0/24
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>
99.0.0.0/24, ubest/mbest: 1/0
*via 1.1.1.6, [20/0], 14:34:51, bgp-100, external, tag 200
via 2.2.2.6, [200/0], 14:34:51, bgp-100, internal, tag 200 (backup)
```

UFIB output:

```
switch# show forwarding route 123.1.1.0 detail module 8
Prefix 123.1.1.0/24, No of paths: 1, Update time: Wed Jul 11 19:00:12 2018
Vobj id: 141   orig_as: 65002   peer_as: 65100  rnh: 10.3.0.2
10.4.0.2           Ethernet8/4         DMAC: 0018.bad8.4dfd
packets: 2  bytes: 3484   Repair path  10.3.0.2   Ethernet8/3 DMAC:
0018.bad8.4dfd packets: 0
bytes: 1
```

Configure BGP PIC Core

Follow these steps to configure BGP PIC Core.

Procedure

Step 1 **configure terminal**

Example:

```
switch# configure terminal
```

Enter global configuration mode.

Step 2 **[no] system pic-core**

Example:

```
switch(config)# system pic-core
```

Manage PIC enable.

Step 3 **copy running-config startup-config**

Example:

```
switch(config)# copy running-config startup-config
```

Saves this configuration change.

Step 4 **reload**

Example:

```
switch(config)# reload
```

Reboots the entire device.

Clear 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 or IPv6 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 or IPv6 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** | **ipv6**} {**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** | **ipv6**} {**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.

Command	Purpose
clear bgp {**ipv4** | **ipv6** } {*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 or IPv6 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 or IPv6 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 {**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 or IPv6 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 or IPv6 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 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 address of a neighbor. • *ip-prefix* —IPv4. 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 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 address of a neighbor. • *ip-prefix* —IPv4. 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 mbgp {**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 or IPv6 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 or IPv6 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 mbgp 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 address of a neighbor. • *ip-prefix* —IPv4. 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 mbgp flap-statistics [*ip-neighbor* | *ip-prefix*] [**vrf** *vrf-name*]	Clears route flap statistics in one or more networks. The arguments are as follows: • *ip-neighbo*r—IPv4 address of a neighbor. • *ip-prefix* —IPv4. 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.

Verify the Basic BGP Configuration

To display the BGP configuration, 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 {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix* **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*] {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **community-list** *list-name* [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP community list.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix* **extcommunity** [**regexp** *expression* | [**generic** [**non-transitive** | **transitive**] *aa4:nn* [**exact-match**]} [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP extended community.

Command	Purpose
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix* **extcommunity-list** *list-name* [**exact-match**]} [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP extended community list.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix* {**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 {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix* **history-paths** [**regexp** *expression*] [**vrf** *vrf-name*]	Displays the BGP route history paths.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix* **filter-list** *list-name* [**vrf** *vrf-name*]	Displays the information for the BGP filter list.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **neighbors** [*ip-address* | *ipv6-prefix*] [**vrf** *vrf-name*]	Displays the information for BGP peers. Use the **clear bgp neighbors** command to clear these neighbors.
show bgp {**ipv4** | **ipv6**} **unicast neighbors** [*ip-address* | *ipv6-prefix*] { [**advertised-routes** |**received-routes**] } [**detail**] [**vrf** *vrf-name*] **show bgp** {**ipv4** | **ipv6**} **unicast neighbors** [*ip-address* | *ipv6-prefix*] [**routes**] { [**advertised** |**received**] } [**detail**] [**vrf** *vrf-name*]	Displays the detailed information of all routes: • received from the peer before evaluating inbound route map. • advertised to the peer before updating attributes by outbound route map.
show bgp {**ipv4** | **ipv6**} **unicast neighbors** [*ip-address* | *ipv6-prefix*] [**routes**] [**detail**] [**vrf** *vrf-name*]	Displays the detailed information of all routes received from this peer after evaluating inbound route map.
show bgp {**ipv4** | **ipv6**} **unicast neighbors** [*ip-address* | *ipv6-prefix*] [**advertised-routes** **processed**] [**vrf** *vrf-name*]	Displays brief information of all routes advertised to the peer after updating path attributes by outbound route map with **processed** option.
show bgp {**ipv4** | **ipv6**} **unicast neighbors** [*ip-address* | *ipv6-prefix*] [**advertised-routes** **processed**] [**detail**] [**vrf** *vrf-name*]	Displays detailed information of all routes advertised to the peer after updating path attributes by outbound route map with **processed** option.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **neighbors** [*ip-address* | *ipv6-prefix*] {**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 {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **policy** *name* [**vrf** *vrf-name*]	Displays the BGP policy information. Use the **clear bgp polic** y command to clear the policy information.

Command	Purpose
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **prefix-list** *list-name* [**vrf** *vrf-name*]	Displays the BGP routes that match the prefix list.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **received-paths** [**vrf** *vrf-name*]	Displays the BGP paths stored for soft reconfiguration.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **regexp** *expression* [**vrf** *vrf-name*]	Displays the BGP routes that match the AS_path regular expression.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **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*] show bgp peer-session	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 {**ipv** | **ipv6**} **bgp** [*options*]	Displays the BGP status and configuration information.
show {**ipv** | **ipv6**} **mbgp** [*options*]	Displays the BGP status and configuration information.
show running-configuration bgp	Displays the current running BGP configuration.

Monitor BGP Statistics

To display BGP statistics, use the following commands:

Command	Purpose
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **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 statistics	Displays the BGP statistics.

Configuration Examples for Basic BGP

This example shows a basic BGP configuration:

```
switch(config)# feature bgp
switch(config)# router bgp 64496
switch(config-router)# neighbor 2001:ODB8:0:1::55 remote-as 64496
switch(config-router)# address-family ipv6 unicast
switch(config-router-af)# next-hop-self
```

Related topics

The following topics relate to BGP:

- Configure advance BGP
- Configure Route Policy Manager

Where to go next

See Configure advance BGP for details on these features:

- Peer templates
- Route redistribution
- Route maps

Additional References

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

Where to Go Next

See *Configuring Advance BGP*, for details on the following features:

- Peer templates
- Route redistribution
- Route maps

Configure advance BGP

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 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 devices 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.

Beginning with Cisco NX-OS Release 10.5(1)F, Configuring Basic BGP and Configuring Advanced BGP chapters are merged to create Configuring BGP chapter.

Peer templates

BGP peer templates allow you to create blocks of common configuration 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: This 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: This 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: This 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

A route policy is a configuration mechanism that you associate with a BGP peer to control and modify the routes that BGP recognizes and processes. Route policies use route maps to selectively filter, accept, deny, or modify routes during BGP route updates. You can apply route policies to inbound or outbound route updates, allowing granular control over routing granular control over routing behavior.

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.

Undefined Policies

Currently, in BGP (Border Gateway Protocol) configurations, if a route-map is referenced but has not been defined, the system's default behavior is to deny the route. This can lead to unintended route filtering or route loss, especially during configuration transitions or "Config-Replace" operations where a route-map might be temporarily unavailable while the system processes changes.

Beginning with NX-OS Release 10.6(3), users can configure a "permit-by-default" behavior to avoid route loss during configuration changes. When the command **[no] undefined-policy any default-action permit**is enabled, BGP will permit routes by default if the referenced route-map is missing or not yet ready, rather than denying them. This applies to several BGP configurations, including:

- **redistribute**
- **network** commands
- **aggregate-address**(attribute-map)
- Inbound and outbound route-maps

Note BGP also uses route maps for route redistribution, route aggregation, route dampening, and other features. See Configuring Route Policy Manager, on page 523, 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. Typically eBGP peerings need to be over directly connected interfaces so that convergence will be faster when the interface goes down.

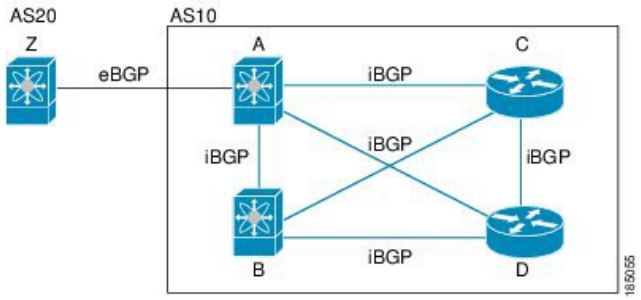
iBGP

Internal BGP (iBGP) is a type of Border Gateway Protocol (BGP) that

- is used to exchange routing information between BGP peers within the same autonomous system (AS)
- used within an AS to distribute external routes learned from other ASes (via eBGP) to all routers within that same AS, typically in multihomed BGP networks that have multiple connections to the same external AS
- ensures all routers have a consistent view of external networks, and
- enforces routing policies for traffic leaving the AS.

The figure shows an iBGP network within a larger BGP network.

Figure 29: iBGP Network



iBGP networks are fully meshed. Each iBGP peer has a direct connection to all other iBGP peers to prevent network loops.

For single-hop iBGP peers with update-source configured under neighbor configuration mode, the peer supports fast external fall-over.

You should use loopback interfaces for establishing iBGP 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, on page 374 section for information on multihop, fast external fallovers, and limiting the size of the AS_path attribute.

Note You should configure a separate interior gateway protocol in the iBGP network.

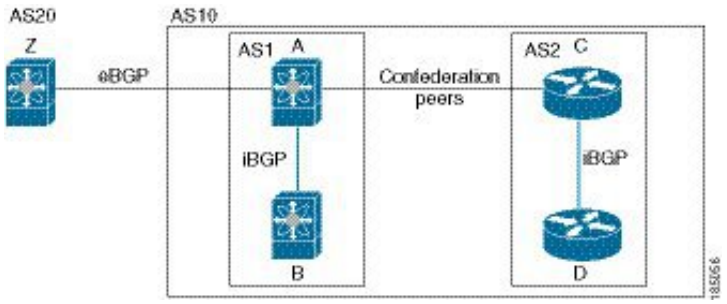
AS confederations

An AS confederation is a BGP architectural technique that divides a large autonomous system (AS) into multiple smaller subautonomous systems, called sub-ASes, which are grouped together to appear as a single AS to external networks. This approach reduces the complexity of a fully meshed iBGP network by limiting the number of iBGP peer connections required within the overall AS.

You can reduce the iBGP mesh by dividing the autonomous system into multiple subautonomous systems and grouping them into a single confederation. A confederation 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 figure shows the BGP network, split into two subautonomous systems and one confederation.

Figure 30: 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.

Route reflectors

Route reflectors are a BGP architectural technique that reduces the complexity of a fully meshed iBGP network by allowing certain routers, called route reflectors, to redistribute routes learned from one iBGP peer to other iBGP peers. This eliminates the need for every iBGP peer to be directly connected to every other iBGP peer.

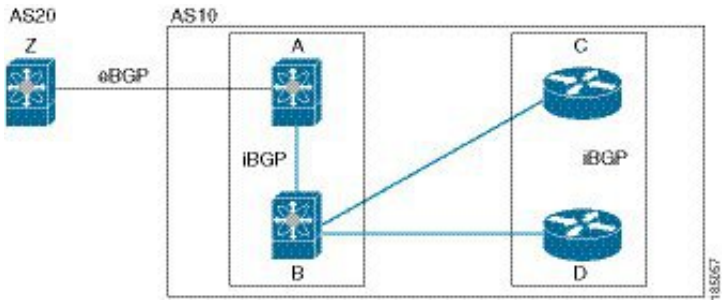
Reflectors pass learned routes to neighbors so that all iBGP peers do not need to be fully meshed.

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 sp iBGP peers do not need to be fully meshed.

The figure shows a simple iBGP configuration with four meshed iBGP speakers (routers 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.

In the 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 31: 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 that are 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 attempts a new session to the peer without capabilities negotiation if you have configured the address family as IPv4. Any other multiprotocol configuration (such as IPv6) requires capabilities negotiation.

Route dampening

Route dampening is a BGP feature that minimizes the propagation of flapping routes across an internetwork. It works by assigning a penalty to routes that frequently alternate between available and unavailable states (flapping). When the penalty exceeds a configured suppression limit, the route is suppressed (dampened) and not advertised to neighbors until the penalty decays below a reuse threshold.

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.



Note When leaking routes between VRFs with BGP, you may experience that route changes are not reflected in the destination VRF until several minutes later. This can happen due to BGP dampening mechanisms which ignore route changes based on metric changes to avoid cases in which an IGP is flapping (link issues, for example) and you do not want these flaps to be propagated beyond. If you suffer from this condition and you want the routes to be immediately updated on the destination VRF as well, see Configure dampen-igp-metric in BGP on Cisco Nexus Switches.

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.

To configure `as-path multipath-relax` command effectively, configure the command per VRF under BGP. Also, configure as-path multipath-relax command under the custom VRF so that multiple routers get installed in the custom VRF Route-Target (RT).

The BGP best-path algorithm considers the paths as equal-cost paths if these 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. For more information, see the BGP Additional Paths section.

Note Paths that are received from different AS confederations are considered as equal-cost paths if the external AS_path values and the other attributes are identical.

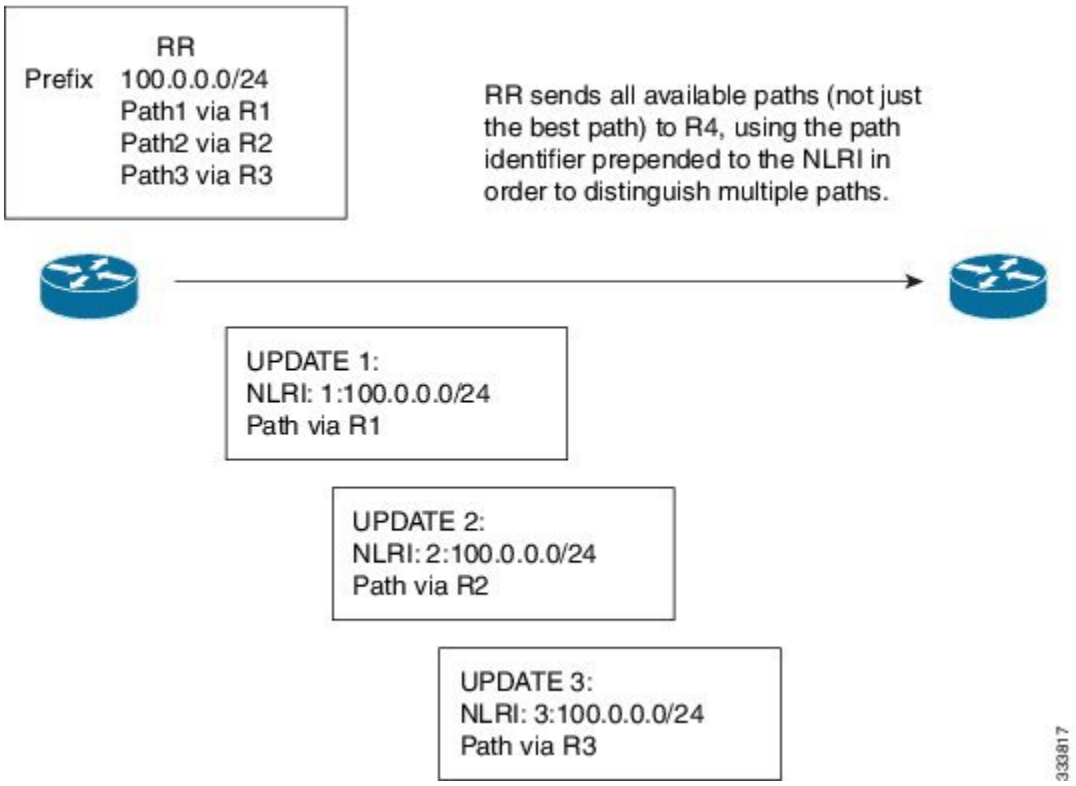
Note When you configure a route reflector for iBGP multipath, and the route reflector advertises the selected best path to its peers, the next hop for the path is not modified.

BGP additional paths

BGP supports the additional paths feature, which allows the BGP speaker to propagate and accept multiple paths for the same prefix without the new paths replacing any previous ones. This feature allows BGP speaker peers to negotiate whether they support advertising and receiving multiple paths per prefix and advertising such paths. A special 4-byte path ID is added to the network layer reachability information (NLRI) to differentiate multiple paths for the same prefix sent across a peer session. Figure Figure 32: BGP Route Advertisement with the Additional Paths Capability, on page 331 illustrates the BGP additional paths capability.

Only one BGP best path is advertised, and the BGP speaker accepts only one path for a given prefix from a given peer. If a BGP speaker receives multiple paths for the same prefix within the same session, it uses the most recent advertisement.

Figure 32: BGP Route Advertisement with the Additional Paths Capability



For information on configuring BGP additional paths, see the Configure BGP Additional Paths, on page 370section.

Route aggregation

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.

A summary entry is created in the BGP table when you issue the **aggregate-address** command, but the summary entry is not eligible for advertisement until a subset of the aggregate is found in the table.

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 the Configuring 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 Routing Information Base (RIB) that may affect BGP next-hop reachability.

BGP receives notifications from the RIB when the next-hop information changes (event-driven notifications). BGP is notified when any of these events occurs:

- The next hop becomes unreachable.
- The next hop becomes reachable.
- The fully recursed Interior Gateway Protocol (IGP) metric to the next hop changes.
- The first hop IP address or first hop interface changes.
- The next hop becomes connected.
- The next hop becomes unconnected.
- The next hop becomes a local address.
- The 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 next-hop reachability, such as the loss of next hops resulting in a switchover to a different path. A change in the IGP metric for a next hop resulting in a switchover to a different path can also be considered a critical event.
- Non-critical events are related to next hops being added without affecting the best path or changing the IGP metric to a single next hop.

See Configuring BGP Next-Hop Address Tracking for more information.

Route redistribution

Route redistributions are configurations in BGP that allow you to import routes from static routes or other routing protocols into BGP. To control which routes are redistributed, you must use a route map. A route map acts as a filter and can match routes based on attributes such as destination prefix, originating protocol, route type, route tag, and more. This filtering helps ensure that only desired routes are passed into BGP. See Configuring Route Policy Manager, on page 523, for more information.

You can also use route maps to override the default behavior of route redistribution. However, caution is necessary because incorrect use of route maps can cause network loops, which can destabilize the network.

Theses examples show how to use route maps to change the default behavior.

You can change the default behavior for scenario 1 by modifying the route map as follows:

```
route-map foo permit 10
match route-type internal
router ospf 1
redistribute bgp 100 route-map foo
```

Similarly, you can change the default behavior for scenario 2 by modifying the route map as follows:

```
route-map foo deny 10
match route-type internal
router ospf 1
vrf bar
redistribute bgp 100 route-map foo
```

Labeled and unlabeled unicast routes

In release 7.0(3)I7(6), SAFI-1 (unlabeled unicast) and SAFI-4 (labeled unicast routing) are now supported for IPv4 BGP on a single session. For more information, see the *Cisco Nexus 9000 Series NX-OS Label Switching Configuration Guide, Release 7.x*.

BFD

Bidirectional forwarding detection (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. Advanced BGP supports BFD for IPv4 and IPv6.

BFD for BGP is supported on eBGP peers and iBGP single-hop peers. Configure the **update-source** option in neighbor configuration mode for iBGP single-hop peers using BFD.

Beginning with Cisco NX-OS Release 9.3(3), BFD for BGP is also supported for BGP IPv4 and IPv6 prefix peers. This support enables BGP to use multihop BFD, which improves BGP convergence times. Both single-hop and multihop BGP are supported for prefix peers.

Beginning with Cisco NX-OS Release 9.3(3), BFD supports BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families. However, BFD multihop is not supported with unnumbered BGP.

See the Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide for more information.

Tune BGP

You can modify the default behavior of BGP through BGP timers and by adjusting the best-path algorithm.

BGP timers

BGP timers are protocol timers used in the Border Gateway Protocol (BGP) that

- control the timing of periodic keepalive messages sent between BGP peers,
- determine the timeout period for detecting lost neighbor sessions when keepalives are not received, and
- manage other specific protocol events and features.

These timers are typically configured in seconds and include a random adjustment to prevent synchronization of timer events across different BGP peers.

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. The timers include a random adjustment so that the same timers on different BGP peers trigger at different times.

Tune 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 (MED) 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, one set of routes for IPv4 multicast routing, and one set of routes for IPv6 multicast routing. You can use MP-BGP for reverse-path forwarding (RPF) checks in IP multicast networks.

Multicast BGP does not propagate multicast state information, so 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, all routers can forward IPv4 and IPv6 traffic. Therefore, 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).

Beginning with Cisco NX-OS Release 9.2(2), Cisco Nexus 9500 platform switches with -R line cards support RFC 5549.

Currently, NX-OS does not support IPv6 recursive next-hops (RNH) for an IPv4 route.

RFC 6368

Introduction

Cisco NX-OS RFC 6368 enables Internal Border Gateway Protocol (iBGP) support between Provider Edge (PE) and Customer Edge (CE) feature is implemented in Cisco NX-OS.

In current deployments, when BGP is used as the PE or CE routing protocol, these peering sessions are configured as an external peering between the VPN provider autonomous system (AS) and the customer network autonomous system.

RFC 6368 adds support for these peers to be configured as iBGP peers instead.

Beginning with Cisco NX-OS Release 10.1(2), RFC 6368 support is enabled for EVPN-VxLANv4 and EVPN-VxLANv6.

Framework

Beginning with Cisco NX-OS Release 10.1(2), you can use iBGP PE-CE to do these:

- You can have one single Autonomous System Number (ASN) on the multiple sites of the VRF, without the deployment of External Border Gateway Protocol (eBGP) with as-override.
- You can give internal route reflection towards the CE routers, acting as if the Provider core is one transparent Route Reflector (RR).

With this feature, the VRF sites can have the same ASN as the provider core. However, in case the ASN of the VRF sites are different than the ASN of the provider core, it can be made to appear the same with the use of the feature local Autonomous System (AS).

Implement iBGP PE-CE

Here are the two major parts to make this feature work:

- A new attribute `ATTR_SET` added to the BGP protocol to carry the VPN BGP attributes across the provider core in a transparent manner.
- Make the PE router a RR for the iBGP sessions towards the CE routers in the VRF.

The new `ATTR_SET` attribute allows the provider to carry all the BGP attributes of the customer transparently and does not interfere with the provider attributes and BGP policies. Such attributes are the cluster list, local preference, and so on.

BGP customer route attribute

`ATTR_SET` is the new BGP attribute used to carry the VPN BGP attributes of the provider customer. It is an optional transitive attribute. In this attribute, Local Preference, Med, Origin, AS Path, Originator ID, Cluster list attributes will be carried across the provider network. The `ATTR_SET` attribute has the format:

```
+------------------------------+
| Attr Flags (O|T) Code = 128  |
+------------------------------+
| Attr. Length (1 or 2 octets) |
+------------------------------+
| Origin AS (4 octets)         |
```

```
+-----------------------------+
|Path Attributes (variable)   |
+-----------------------------+
```

- Attribute Flags are regular BGP attribute flags.
- Attribute length indicates whether the length is one or two octets.
- Origin AS field is to prevent a leak of one route that originated in one AS to be leaked to another AS without proper manipulation of the `AS_PATH`.
- The variable-length path attributes field carries VPN BGP attributes that must be carried across the provider core.

For more information on the implementation of iBGP PE-CE, see IOS Implementation of the iBGP PE-CE Feature.

This example shows BGP neighbor configuration on PE device for iBGP Customer Edge device:

```
router bgp 200
                vrf nxbgp3-leaf2-2
                address-family ipv4 unicast
                redistribute static route-map ALLOW-ALL
                address-family ipv6 unicast
                redistribute static route-map ALLOW-ALL
                neighbor 101.101.101.101 remote-as 200
                description ibgp sample config
                internal-vpn-client (1)
                address-family ipv4 unicast
                route-reflector-client (2)
                next-hop-self (3)
```

BGP monitoring protocol

BGP Monitoring Protocol (BMP) monitors BGP updates and peer statistics and is supported for all Cisco Nexus 9000 Series switches.

Using this protocol, the BGP speaker connects to external BMP servers and sends them information regarding BGP events. A maximum of two BMP servers can be configured in a BGP speaker, and each BGP peer can be configured for monitoring by all or a subset of the BMP servers. The BGP speaker does not accept any information from the BMP server.

Graceful restart and high availability

Cisco NX-OS supports nonstop forwarding and graceful restart for BGP.

You can use nonstop forwarding (NSF) for BGP to forward data packets along known routes in the Forward Information Base (FIB) while the BGP routing protocol information is being restored following a failover. With NSF, BGP peers do not experience routing flaps. During a failover, the data traffic is forwarded through intelligent modules while the standby supervisor becomes active.

If a Cisco NX-OS router experiences a cold reboot, the network does not forward traffic to the router and removes the router from the network topology. In this scenario, BGP experiences a nongraceful restart and removes all routes. When Cisco NX-OS applies the startup configuration, BGP reestablishes peering sessions and relearns the routes.

A Cisco NX-OS router that has dual supervisors can experience a stateful supervisor switchover. During the switchover, BGP uses nonstop forwarding to forward traffic based on the information in the FIB, and the

system is not removed from the network topology. A router whose neighbor is restarting is referred to as a "helper." After the switchover, a graceful restart operation begins. When it is in progress, both routers reestablish their neighbor relationship and exchange their BGP routes. The helper continues to forward prefixes pointing to the restarting peer, and the restarting router continues to forward traffic to peers even though those neighbor relationships are restarting. When the restarting router has all route updates from all BGP peers that are graceful restart capable, the graceful restart is complete, and BGP informs the neighbors that it is operational again.

BGP needs to converge before graceful-restart timer expires. BGP graceful-restart timer needs to be increased in high route scale network accordingly in order to avoid temporary traffic loss. If BGP itself provides the reachability to open other BGP sessions, then stalepath-time should also be increased to accommodate for the extra time needed to converge the overlay session after the initial underlay session has already converged.

When a router detects that a graceful restart operation is in progress, both routers exchange their topology tables. When the router has route updates from all BGP peers, it removes all the stale routes and runs the best-path algorithm on the updated routes.

After the switchover, Cisco NX-OS applies the running configuration, and BGP informs the neighbors that it is operational again.

For single-hop iBGP peers with update-source configured under neighbor configuration mode, the peer supports fast external fall-over.

Beginning with Cisco NX-OS Release 9.3(3), BGP prefix peers support graceful restarts.

With the additional BGP paths feature, if the number of paths advertised for a given prefix is the same before and after restart, the choice of path ID guarantees the final state and removal of stale paths. If fewer paths are advertised for a given prefix after a restart, stale paths can occur on the graceful restart helper peer.

Low memory handling

BGP low memory handling is a mechanism that manages BGP peer sessions and memory usage when the router experiences low memory conditions. It helps maintain stability and performance by controlling peer establishment and shutdown based on the severity of the memory alert.

BGP reacts to low memory for theses conditions:

- Minor alert: BGP does not establish any new eBGP peers. BGP continues to establish new iBGP peers and confederate peers. Established peers remain, but reset peers are not re-established.
- Severe alert: BGP shuts down select established eBGP peers every two minutes until the memory alert becomes minor. For each eBGP peer, BGP calculates the ratio of total number of paths received to the number of paths selected as best paths. The peers with the highest ratio are selected to be shut down to reduce memory usage. You must clear a shutdown eBGP peer before you can bring the eBGP peer back up to avoid oscillation.



Note You can exempt important eBGP peers from this selection process.

- Critical alert: BGP gracefully shuts down all the established peers. You must clear a shutdown BGP peer before you can bring the BGP peer back up.

See the Tuning BGP section for more information on how to exempt a BGP peer from a shutdown due to a low memory condition.

Virtualization support

You can configure one BGP instance. BGP supports virtual routing and forwarding (VRF) instances.

Enable 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

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 forward**

Example:

```
switch(config-if)# ip forward
```

Allows IPv4 traffic on the interface even when there is no IP address configuration on that interface.

Step 4 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if)# copy running-config startup-config
```

Saves this configuration change.

Configure BGP Session Templates

You can use BGP session templates to simplify the 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

You must enable BGP (see the Enabling BGP 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.
- When using BGP Peer Template, there is no check for the commands used inside template to verify if that command applies to iBGP/eBGP peer or not. For example if you create a template and add a command "Remove-private-as" inside a template and then assign this tempate to iBGP peer, then no error will be printed saying this command "Remove-private-as" does not apply to iBGP peer.

Procedure

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 65535
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 clear text 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 65535
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:

```
switch(config-router-neighbor)# show bgp
peer-session BaseSession
```

Displays the peer-policy template.

Step 11 (Optional) **copy running-config startup-config**

Example:

```
switch(config-router-neighbor)# copy
running-config startup-config
```

Saves this configuration change.

Use the **show bgp neighbor** command to see the template applied.

Example

This example shows how to configure a BGP peer-session template and apply it to a BGP peer:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router bgp 65536</userinput>
switch(config-router)# <userinput>template peer-session BaseSession</userinput>
switch(config-router-stmp)# <userinput>timers 30 90</userinput>
switch(config-router-stmp)# <userinput>exit</userinput>
switch(config-router)# neighbor <userinput>192.168.1.2 remote-as 65536</userinput>
switch(config-router-neighbor)# <userinput>inherit peer-session BaseSession</userinput>
switch(config-router-neighbor)# <userinput>description Peer Router A</userinput>
switch(config-router-neighbor)# <userinput>address-family ipv4 unicast</userinput>
switch(config-router-neighbor-af)# <userinput>copy running-config startup-config</userinput>
```

Configure 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.

Use the **show bgp neighbor** command to see the template applied. See the *Cisco Nexus 9000 Series NX-OS Unicast Routing Command Reference*, for details on all commands available in the template.

Before you begin

You must enable BGP (see the Enabling BGP 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

Step 1 **configure terminal**

Example:

```
switch# configure terminal
```

Enters configuration mode.

Step 2 **router bgp***autonomous-system-number*

Example:

```
switch(config)# router bgp 65535
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-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 65535
switch(config-router-neighbor)#
```

Places the router in the 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 address family specified.

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.

Use the **show bgp neighbor** command to see the template applied.

Example

This example shows how to configure a BGP peer-policy template and apply it to a BGP peer:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router bgp 65536</userinput>
switch(config-router)# <userinput>template peer-session BasePolicy</userinput>
switch(config-router-ptmp)# <userinput>maximum-prefix 20</userinput>
switch(config-router-ptmp)# <userinput>exit</userinput>
switch(config-router)# <userinput>neighbor 192.168.1.1 remote-as 65536</userinput>
switch(config-router-neighbor)# <userinput>address-family ipv4 unicast</userinput>
switch(config-router-neighbor-af)# <userinput>inherit peer-policy BasePolicy</userinput>
switch(config-router-neighbor-af)# <userinput>copy running-config startup-config</userinput>
```

Configure 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

You must enable BGP (see the Enabling BGP 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

Step 1 **configure terminal**

Example:

```
switch# configure terminal
```

Enters global configuration mode.

Step 2 **router bgp***autonomous-system-number*

Example:

```
switch(config)# router bgp 65535
```

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
```

Enters peer template configuration mode.

Step 4 (Optional) **inherit peer-session***template-name*

Example:

```
switch(config-router-neighbor)# inherit
peer-session BaseSession
```

Adds a peer-session template to 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-policy***template-name*

Example:

```
switch(config-router-neighbor-af)#
inherit peer-policy BasePolicy 1
```

Applies a peer-policy template to the neighbor address family configuration.

Step 7 **exit**

Example:

```
switch(config-router-neighbor-af)# exit
```

Exits BGP neighbor address family configuration mode.

Step 8 (Optional) **timers** *keepalive hold*

Example:

```
switch(config-router-neighbor)# timers
45 100
```

Adds the BGP timer values to the peer.

These values override the timer values in the peer-session template, BaseSession.

Step 9 **exit**

Example:

```
switch(config-router-neighbor)# exit
```

Exits BGP neighbor configuration mode.

Step 10 **neighbor** *ip-address* **remote-as** *as-number*

Example:

```
switch(config-router)# neighbor
192.168.1.2 remote-as 65535
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)# show
bgp peer-template BasePeer
```

Displays the peer template.

Step 14 (Optional) **copy running-config startup-config**

Example:

```
switch(config-router-neighbor)# copy
running-config startup-config
```

Saves this configuration change.

Use the **show bgp neighbor** command to see the template applied.

Example

This example shows how to configure a BGP peer template and apply it to a BGP peer:

```
switch# <userinput>configure terminal</userinput>
switch(config)# <userinput>router bgp 65536</userinput>
switch(config-router)# <userinput>template peer BasePeer</userinput>
switch(config-router-neighbor)# <userinput>inherit peer-session BaseSession</userinput>
switch(config-router-neighbor)# <userinput>address-family ipv4 unicast</userinput>
switch(config-router-neighbor-af)# <userinput>inherit peer-policy BasePolicy 1</userinput>
switch(config-router-neighbor-af)# <userinput>exit</userinput>
switch(config-router-neighbor)# <userinput>exit</userinput>
switch(config-router)# <userinput>neighbor 192.168.1.2 remote-as 65536</userinput>
switch(config-router-neighbor)# <userinput>inherit peer BasePeer</userinput>
switch(config-router-neighbor)# <userinput>copy running-config startup-config</userinput>
```

Configure 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.

Procedure

Step 1 **timers prefix-peer-timeout** *value*

Example:

```
switch(config-router-neighbor)# timers
prefix-peer-timeout 120
```

Configures the BGP prefix peering timeout value in router configuration mode. 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.

Step 2 **maximum-peers** *value*

Example:

```
switch(config-router-neighbor)#
maximum-peers 120
```

Configures the maximum number of peers for this prefix peering in neighbor configuration mode. The range is from 1 to 1000.

Example

This example shows how to configure a prefix peering that accepts up to 10 peers:

```
switch(config)# <userinput>router bgp 65536</userinput>
switch(config-router)# <userinput>timers prefix-peer-timeout 120</userinput>
switch(config-router)# <userinput>neighbor 10.100.200.0/24 remote-as 65536</userinput>
switch(config-router-neighbor)# <userinput>maximum-peers 10</userinput>
switch(config-router-neighbor)# <userinput>address-family ipv4 unicast</userinput>
switch(config-router-neighbor-af)#
```

Use the **show bgp ipv4 unicast 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.

Configure BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families

You can configure BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families for automatic BGP neighbor discovery using unnumbered interfaces. Doing so allows you to set up BGP sessions using an interface name as a BGP peer (rather than interface-scoped addresses). This feature relies on ICMPv6 neighbor discovery (ND) route advertisement (RA) for automatic neighbor discovery and on RFC 5549 for sending IPv4 routes with IPv6 next hop.

Before you begin

You must enable BGP (see the Enabling BGP section).

Procedure

Step 1 **configure terminal**

Example:

```
switch# configure terminal
```

Enters configuration mode.

Step 2 **router bgp** *autonomous-system-number*

Example:

```
switch(config)# router bgp 65535
                        switch(config-router)#
```

Enables BGP and assigns the autonomous system 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** *interface-name* **remote-as** {*as-number* | **route-map** *map-name*}

Example:

```
switch(config-router)# neighbor
                       Ethernet1/1 remote-as 65535
                       switch(config-router-neighbor)#
```

Places the router in the neighbor configuration mode for BGP routing and configures the interface for BGP peering.

Note You can specify only Ethernet interfaces, port-channel interfaces, subinterfaces, and breakout interfaces.

Beginning with Cisco NX-OS Release 9.3(6), you can specify a route map, which can contain AS lists and ranges. See *Dynamic AS Numbers for Prefix Peers and Interface Peers* for more information about using dynamic AS numbers.

interface-name can be a range if the configuration needs to be applied to more than one interface.

Step 4 **inherit peer** *template-name*

Example:

```
switch(config-router-neighbor)# inherit peer PEER
```

Inherits the peer template.

Step 5 **address-family** {**ipv4** | **ipv6**} **unicast**

Example:

```
switch(config-router-neighbor)#
                       address-family ipv4 unicast
                       switch(config-router-neighbor-af)#
```

Enters global address family configuration mode for the address family specified.

Step 6 (Optional) **show bgp** {**ipv4** | **ipv6**} **unicast neighbors** *interface*

Example:

```
switch(config-router-neighbor-af)# show
                       bgp ipv4 unicast neighbors e1/25
```

Example:

```
switch(config-router-neighbor-af)# show
                       bgp ipv6 unicast neighbors 3FFE:700:20:1::11
```

Displays information about BGP peers.

Step 7 (Optional) **show ip bgp neighbors** *interface-name*

Example:

```
switch(config-router-neighbor-af)# show ip bgp neighbors Ethernet1/1
```

Displays the interface used as a BGP peer.

Step 8 (Optional) **show ipv6 routers** [**interface** *interface*]

Example:

```
switch(config-router-neighbor-af)# show ipv6 routers interface Ethernet1/1
```

Displays the link-local address of remote IPv6 routers, which is learned through IPv6 ICMP router advertisement.

Step 9 (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 Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families.

iBGP Interface Peering Configuration for Leaf 1:

```
switch# configure terminal
                switch(config)# router bgp 65000
                switch(config-router)# neighbor Ethernet1/1 remote-as 65000
                switch(config-router-neighbor)# inherit peer PEER
                switch(config-router-neighbor)# address-family ipv4 unicast
                switch(config-router-neighbor)# address-family ipv6 unicast
                switch(config-router-neighbor-af)# copy running-config startup-config
```

This example shows sample output for BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families:

```
switch(config-router-neighbor)# show bgp ipv4 unicast neighbors e1/15.1
                BGP neighbor is fe80::2, remote AS 100, ibgp link, Peer index 4
                Peer is an instance of interface peering Ethernet1/15.1
                BGP version 4, remote router ID 5.5.5.5
                Neighbor previous state = OpenConfirm
                BGP state = Established, up for 2d16h
                Neighbor vrf: default
                Peer is directly attached, interface Ethernet1/15.1
                Last read 00:00:54, hold time = 180, keepalive interval is 60 seconds
                Last written 00:00:08, keepalive timer expiry due 00:00:51
                Received 3869 messages, 0 notifications, 0 bytes in queue
                Sent 3871 messages, 0 notifications, 0(0) bytes in queue
                Enhanced error processing: On
                0 discarded attributes
                Connections established 2, dropped 1
                Last reset by peer 2d16h, due to session closed
                Last error length received: 0
                Reset error value received 0
                Reset error received major: 104 minor: 0
                Notification data received:
                Last reset by us never, due to No error
                Last error length sent: 0
                Reset error value sent: 0
                Reset error sent major: 0 minor: 0
                --More--
```

Interface Configuration:

IPv6 needs to be enabled on the corresponding interface using one of the following commands:

- **ipv6 address** *ipv6-address*

- **ipv6 address use-link-local-only**
- **ipv6 link-local** *link-local-address*

```
switch# configure terminal
                switch(config)# interface Ethernet1/1
                switch(config-if)# ipv6 address use-link-local-only
```



Note If an IPv4 address is not configured on the interface, the **ip forward** command must be configured on the interface to enable IPv4 forwarding.



Note IPv6 ND timers can be tuned to speed up neighbor discovery and for BGP faster route convergence.

```
switch(config-if)# ipv6 nd ra-interval 4 min 3
                   switch(config-if)# ipv6 nd ra-lifetime 10
```



Note Beginning with Cisco NX-OS Release 9.3(6), for customer deployments with parallel links, the following command must be added in interface mode:

```
switch(config-if)# ipv6 link-local use-bia
```

The command makes IPv6 LLA unique across different interfaces.



Note Beginning with Cisco NX-OS Release 10.6(3), you can configure DLB and BGP Unnumbered Link Local IPv6 with WCMP.

- Example describes the configuration:

```
interface Ethernet1/1
ipv6 link-local use-bia
ip forward
```

- Example describes the DMZ link bandwidth with IPv6

```
 route-map link_bandwidth_x permit 10
    set extcommunity bandwidth n
 harware profile dlb
 ..
router bgp <asn>
neighbor Ethernet <x>
remote-as <asn> address-family ipv6 unicast
    send-community extended
    route-map <link_bandwidth_x> in
    dmz-link-bandwidth
```

Configure BGP Authentication

You can configure BGP to authenticate route updates from peers using MD5 digests.

Alternatively, beginning with Cisco NX-OS Release 10.4(2)F, you can configure BGP to authenticate route updates from peers using TCP Authentication Option (TCP AO).

Beginning with Cisco NX-OS Release 10.3(3)F, Type-6 encryption for BGP password is supported on Cisco NX-OS switches. Following encryption types are supported:

- AES based encryption
- A configurable encryption-key called as primary-key is used for encryption and decryption of secrets.

To configure BGP to use MD5 digests or TCP AO, use the following command in neighbor configuration mode:

Before you begin

- Ensure the primary-key is configured using the **key config-key ascii** <*primary_key*> command on Cisco NX-OS switches.
- For Type-6 encryption to function properly, ensure **feature password encryption aes** is enabled on Cisco NX-OS switches.
- See Configuring TCP Authentication Option to configure and use TCP keychain authentication option for BGP neighbor session authentication.

Procedure

Step 1 **key config-key ascii** <*primary_key*>

Example:

```
switch# key config-key ascii 0123456789012345
```

Configures the primary-key.

Note
- Enter this command only if the primary key is not configured.
- If the primary key is already configured and if you enter this command, you are actually modifying the existing primary-key value. To modify to the new value, enter the existing primary-key value when prompted.

Step 2 **configure terminal**

Example:

```
switch# configure terminal
```

Enters global configuration mode.

Step 3 **feature password encryption aes**

Example:

```
switch(config)# feature password encryption aes
```

Enables the AES password encryption.

Step 4 **router bgp** *AS number*

Example:

```
switch(config-router)# router bgp 1
```

Enters to BGP router mode.

Step 5 **template peer***template name*

Example:

```
switch(config-router-neighbor)# template peer abc
```

Enters to BGP neighbor mode.

Step 6 **password** {**0** | **3** | **7** | **6**} *string*

Example:

```
switch(config-router-neighbor)# password 6
JDYk51Jok2oVh7IaM/yC02kc3yTrQAc/I2lca1mKPC4P+Nljx/eR6wyngnJWInHTpHwxsCQ40e1P90tnk8I1QDuTsc3XYGLTOAA=
```

Configures an MD5 password for BGP neighbor sessions.

Note When you configure the Type-0/Type-3/Type-7 newly, if primary-key is configured and then if **feature password encryption aes** is enabled, the Type-0/3/7 is automatically encrypted to the Type-6 password.

Step 7 (Optional) **encryption re-encrypt obfuscated**

Example:

```
switch# encryption re-encrypt obfuscated
```

Encrypts the existing Type-0/Type-3/Type-7 password to Type-6 password.

Step 8 (Optional) **encryption delete type-6**

Example:

```
switch# encryption delete type-6
```

Deletes the Type-6 encrypted password.

Step 9 (Optional) **ao** <*Keychain-name*> [**include-tcp-options**]

Configures option to specify whether the TCP option headers (other than TCP AO option) will be included while computing the MAC digest of the packets.

Configure TCP authentication option

This document describes how to configure TCP authentication option on Cisco NX-OS devices.

About TCP authentication option

With TCP Authentication Option (TCP-AO), defined in RFC 5925, you can protect long-lived TCP connections against replays using stronger Message Authentication Codes (MACs).

TCP-AO is the proposed replacement for TCP MD5, defined in RFC 2385. Unlike TCP MD5, TCP-AO is resistant to collision attacks, provides algorithmic agility and support for key management.

TCP-AO has the following distinct features:

- TCP-AO supports the use of stronger Message Authentication Codes (MACs) to enhance the security of long-lived TCP connections.
- TCP-AO protects long-lived TCP connections against replays and coordinates key changes between endpoints by providing a more explicit key management.

he TCP-AO feature deprecates TCP MD5. Cisco NX-OS devices will continue to support the TCP-MD5 option for legacy BGP peers. However, a configuration in which one end of the peering is configured with the TCP MD5 option, and the other with the TCP-AO option is not supported.

TCP-AO key chain

TCP-AO is based on traffic keys and Message Authentication Codes (MACs) generated using the keys and a MAC algorithm. The traffic keys are derived from master keys that you can configure in a TCP-AO key chain. Use the **key chain** *key-chain-name* **tcp** command in the global configuration mode to create a TCP-AO key chain and configure keys in the chain. The TCP-AO key chain must be configured on both the peers communicating via a TCP connection.

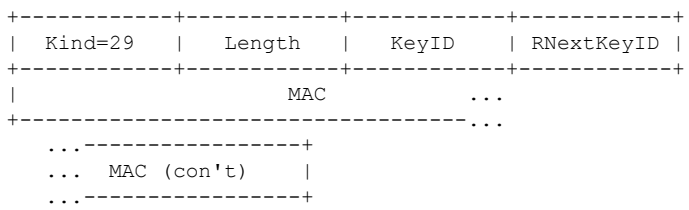
Keys in a TCP-AO key chain have the following configurable properties:

Table 25: TCP-AO key chain

Configurable Property	Description
send-id	Key identifier of the TCP-AO option of the outgoing segment. The send identifier configured on a router must match the receive identifier configured on the peer.
recv-id	Key identifier compared with the TCP-AO key identifier of the incoming segment during authentication. The receive identifier configured on a router must match the send identifier configured on the peer.
cryptographic-algorithm	The MAC algorithm to be used to create MACs for outgoing segments. The algorithm can be one of the following: • AES-128-CMAC authentication algorithm • HMAC-SHA-1 authentication algorithm • HMAC-SHA-256 authentication algorithm

Configurable Property	Description
include-tcp-options	This flag indicates whether TCP options other than TCP-AO will be used to calculate MACs. With this flag enabled, the contents of all options along with a zero-filled authentication option, is used to calculate the MAC. When the flag is disabled, all options other than TCP-AO are excluded from MAC calculations. This flag is disabled by default. **Note** The configuration of this flag is overridden by the application configuration when the application configuration is available.
send-lifetime	This configuration determines the time for which a key is valid and can be used for TCP-AO-based authentication of TCP segments. When the lifetime of key elapses and the key expires, the next key with the youngest lifetime is selected.
key-string	The key string is a pre-shared master key configured on both peers and is used to derive the traffic keys.

TCP-AO Format



The fields of the TLV format are as follows:

- Kind: Indicates TCP-AO with a value of 29.
- Length: Indicates the length of the TCP-AO sequence.
- KeyID: The send identifier of the Master Key Tuple (MKT) that was used to generate the traffic keys.
- RNextKeyID: The receive identifier of the MKT that is ready to be used to authenticate received segments.
- MAC: The MAC computed for the TCP segment data and the prefixed pseudo header.

Master key tuples

Traffic keys are the keying material used to compute the message authentication codes of individual TCP segments.

Master Key Tuples (MKTs) enable you to derive unique traffic keys, and to include the keying material required to generate those traffic keys. MKTs indicate the parameters under which the traffic keys are configured. The parameters include whether TCP options are authenticated, and indicators of the algorithms used for traffic key derivation and MAC calculation.

Each MKT has the following two identifiers:

- *SendID* : The SendID identifier is inserted as the KeyID identifier of the TCP AO option of the outgoing segments.
- *RecvID* : The RecvID is matched against the TCP AO KeyID of the incoming segments.

TCP-AO key rollover

TCP-AO keys are valid for a defined duration configured using the send-lifetime. If send-lifetime is not configured the key is considered inactive. Key rollover is initiated based on the send lifetimes of keys.

TCP-AO coordinates the use of new MKTs using the RNextKeyID and KeyID field on the TCP-AO option field. For hitless key rollovers, new and old keys in keychain configurations need to have at least 15 minutes of overlap. This is required so that the TCP-AO has enough time to coordinate use of the new MKT.

When key rollover is initiated, one of the peer routers, say Router A, indicates that the rollover is necessary. To indicate that the rollover is necessary, Router A sets the RNextKeyID to the receive identifier (recv-id) of the new MKT to be used. On receiving the TCP segment, the peer router, say Router B, looks up the send identifier (send-id) in its database to find the MKT indicated by the RNextKeyID in the TCP-AO payload. If the key is available and valid, Router B sets the current key to the new MKT. After Router B has rolled over, Router A also sets the current key to the new Primary Key Tuples.

Key rollover is initiated with overlapping send-lifetimes and send-lifetime expiry

If you do not configure a new key that can be activated before the expiry of the current key, the key may time out and expire. Such an expiry can cause retransmissions with the peer router rejecting segments authenticated with the expired key. The connection may fail due to Retransmission Time Out (RTO). When new valid keys are configured and usable, the connection can be re-established.

Guidelines and limitations

Use this reference to review the guidelines and limitations.

Follow the guidelines to configure TCP authentication option:

- The send-id and recv-id of each key in the key chain must be unique. Because send-id and recv-id must be chosen from the range 0 to 255, the TCP-AO key chain can have a maximum of 256 keys.
- Only one keychain can be associated with an application connection. Rollover is always performed within the keys in this keychain.
- If the key in use expires, expect segment loss until a new key that has a valid lifetime is configured on each side and keys rollover.

All the following configurations must be done for a TCP-AO keychain key to be considered active: send-id, recv-id, key-string, send-lifetime and cryptographic-algorithm:

- The key chain software process uses the newest key (the youngest key), as determined by the send-lifetime configuration. If two different keys in the same key chain have the same send-lifetime, the key configured most recently is used instead. Configuring two keys with identical send-lifetimes is not recommended.
- he user must configure at least 15 minutes of overlapping time between two overlapping keys.
- Modifying the configuration of a key in use such as key-string, send-id, recv-id, cryptographic-algorithm or send-lifetime will result in TCP connection flap.
- A keychain's configuration type must match the type it has been linked to within the client protocol. If an attempt is made to mismatch these types, a syslog message is generated to notify the user. For example:

It is not supported if a keychain named keychain_abc is configured as a Macsec keychain but is associated as a TCP keychain with BGP. Similarly, the case where the keychain is first associated with the client (a process known as forward-referencing) and then configured as a different keychain type, is also not supported.

Configure TCP key chain and keys

Before you begin

- Ensure that the key-string, send-lifetimes, cryptographic-algorithm, and ids of keys match on both peers.
- Ensure that the send-id on a router matches the recv-id on the peer router. We recommend using the same id for both the parameters unless there is a need to use separate key spaces.
- The send-id and recv-id of a key cannot be reused for another key in the same key chain.
- The key-string is encrypted and stored in the Type-6 format if the AES password encryption feature is enabled and a primary key is configured. Otherwise, the password will be stored in the Type-7 encrypted format.
- For more details, see Configuring a Primary Key and Enabling the AES Password Encryption Feature.

Procedure

Step 1 Use the **key chain** *name* **tcp** command to enter keychain configuration mode for the keychain that you specified.

Example:

```
switch#configure terminal
switch(config)#key chain bgp-keys tcp
```

Step 2 Use the **key** *key-ID* command to enter key configuration mode for the key that you specified.

Example:

```
switch(config-tcpkeychain)# key 13
```

The *key-ID* argument must be a whole number between 0 and 65535.

Step 3 Use the **send-id** *send-ID* command to specify the send identifier for the key.

Example:

```
switch(config-tcpkeychain-tcpkey)#send-id 2
```

The send-ID must be in the range from 0 to 255 and unique value per key chain.

Step 4 Use the **recv-id** *recv-ID* command to specify the receive identifier for the key.

Example:

```
switch(config-tcpkeychain-tcpkey)#recv-id 2
```

The *recv-ID* must be in the range from 0 to 255 and unique value per key chain.

Step 5 Use the **key-string** *[encryption-type] text-string* command to configure the text string for the key. The text-string argument is alphanumeric, case-sensitive, and supports special characters.

Example:

```
switch(config-tcpkeychain-tcpkey)#key-string 0 AS3cureStr1ng
```

The encryption-type argument can be one of the following values:

- 0—The text-string argument that you enter is unencrypted text. This is the default.
- 6—Beginning with Cisco NX-OS Release 10.3(3)F, the Cisco proprietary (Type-6 encrypted) method is supported on Cisco Nexus 9000 Series platform switches.
- 7—The text-string argument that you enter is encrypted. The encryption method is a Cisco proprietary method. This option is useful when you are entering a text string based on the encrypted output of a **show key chain** command that you ran on another Cisco NX-OS device.

The **key-string** command has limitations on using the following special characters in the *text-string*:

Table 26: Configure TCP key chain and keys

Special Character	Description	Comments
|	Vertical bar or pipe	Unsupported at start of key-string
>	Greater than	Unsupported at start of key-string
\	Backslash	Unsupported start or end of a key-string
(	Left parenthesis	Unsupported at start of key-string
'	Apostrophe	Unsupported at start of key-string
"	Quotation mark	Unsupported at start of key-string
?	Question mark	Supported. However, press **Ctrl-V** before entering a question mark (?).

For more information on the special characters usage in commands, see Understanding the Command-Line Interface section.

Step 6 Use the **[no] cryptographic-algorithm {HMAC-SHA-1 | HMAC-SHA-256 | AES-128-CMAC}** command to specify the the algorithm to be used to compute MACs for TCP segments. You can configure only one cryptographic algorithm for each key.

Example:

```
switch(config-tcpkeychain-tcpkey)#cryptographic-algorithm HMAC-SHA-1
```

Step 7 Use the **send-lifetime [local]** *start-time* **duration** [duration-value | infinite | end-time] command to configure a send lifetime for the key.

Example:

```
switch(config-tcpkeychain-tcpkey)#send-lifetime local 01:01:01 Jan 01 2023 01:01:01 Jan 10 2023
```

By default, the device treats the start-time and end-time arguments as UTC. If you specify the *local* keyword, the device treats these times as local times.

The start-time argument is the time of day and date that the key becomes active.

You can specify the end of the send lifetime with one of the following options:

- *duration* duration-value —The length of the lifetime in seconds. The maximum length is 2147483646 seconds (approximately 68 years).

- *infinite*—The send lifetime of the key never expires.
- end-time —The end-time argument is the time of day and date that the key becomes inactive.

Step 8 Use the **include-tcp-options** command to include the TCP Options of the TCP header.

Example:

```
switch(config-tcpkeychain-tcpkey)#include-tcp-options
```

An optional configuration to specify if the full TCP Options part of the TCP header (other than TCP AO option) needs to be included while computing the 'MAC' digest of the packets.

TCP keychain verification commands

Use this reference to verify the TCP keychain.

Use the following steps to verify the TCP keychain.

Table 27: TCP keychain verification commands

Command	Purpose
show key chain *[name] [detail]*	Displays the keychains configured on the device.

```
switch#show key chain
Key-Chain bgp_keys tcp
Key 2 -- text 7 "070e234f"
send-id 2
recv-id 2
cryptographic-algorithm AES_128_CMAC
send lifetime UTC (08:17:00 May 29 2023)-(08:21:00 May 29 2023)
include-tcp-options
Key 3 -- text 7 "070c2058"
send-id 3
recv-id 4
cryptographic-algorithm HMAC-SHA-1
send lifetime UTC (08:20:00 May 29 2023)-(always valid) [active]
include-tcp-options
Key 12 -- text ""
send lifetime UTC (08:20:00 May 29 2023)-(always valid)
```

Note [active] indicates that the key is valid and active otherwise the key is inactive. In the above example only key 3 is active and usable.

The **show key chain detail** command will explicitly display any active and inactive key(s). In the case of Type 6 encryption, the show key chain detail command will display if the type 6 key-string is decryptable or not. It will also display the newest (youngest) active send key that the client is currently using to authenticate packets.

```
switch#show key chain detail
Key-Chain bgp_keys tcp
Key 1 -- text 6 "JDYk9k4kmaciqaH6Eu2+9C0tmCRl9k7JAMYs/fXGbW1lmHP88PAA=="
Type6 Decryptable: yes
send-id 1
recv-id 1
```

```
cryptographic-algorithm HMAC-SHA-1
send lifetime local (18:15:42 May 15 2023)-(always valid) [active]
include-tcp-options
accept-ao-mismatch
Key 2 -- text 6 "JDYkB+Fs8u3ujRDpFSu4tH6H7iTS45JJA6sKeGsBD0L3HjGDeg9AA=="
Type6 Decryptable: yes
send-id 2
recv-id 2
cryptographic-algorithm AES_128_CMAC
send lifetime local (17:10:47 May 15 2023)-(18:15:42 May 15 2023) [inactive]
youngest active send key: 1
```

Configuration example for a TCP keychain

This example shows how to configure a TCP keychain named bgp_keys. Each key text string is encrypted. The keys have overlapping lifetime configurations:

```
key chain bgp_keys tcp
key 1
send-id 1
recv-id 1
key-string 7 070e234f
send-lifetime 01:00:00 Oct 10 2023 01:00:00 Oct 11 2023
cryptographic-algorithm AES-128-CMAC
key 2
send-id 2
recv-id 2
key-string 7 075e731f
send-lifetime 00:45:00 Oct 11 2023 01:00:00 Oct 12 2023
cryptographic-algorithm HMAC-SHA-256
include-tcp-options
```

Resource Public Key Infrastructure (RPKI)

Resource Public Key Infrastructure (RPKI) is a globally distributed database that maps BGP prefixes to their authorized origin-AS numbers, enabling routers to validate the origin of BGP paths.

- RPKI contains Route-Origin-Attestation (ROA) objects that provide a mapping between BGP prefix-blocks and authorized AS numbers.
- Routers connect to RPKI caches to download validation data and build a local RPKI database.
- One RPKI cache can serve multiple routers, and one router can connect to multiple RPKI caches.

RPKI Reference Information

RPKI enables BGP routers to validate the origin-AS of BGP paths by connecting to RPKI caches and using ROA objects for validation.

- RPKI is essential for securing BGP routing by ensuring only authorized AS numbers originate specific prefixes.
- ROA objects are aggregated from multiple RPKI caches to form the RPKI database used by BGP.

1. Configure origin-AS validation on an eBGP router.
2. Use Origin Validation State Extended Community to signal validation state to eBGP peers.

Comparison of RPKI cache and router connectivity:

Table 28: RPKI Cache and Router Connectivity Comparison

Attributes	RPKI Cache	Router
Connectivity	Can serve multiple routers	Can connect to multiple RPKI caches
Role	Provides origin-AS validation data	Downloads validation data for BGP

Note TIP: Beginning with Cisco NX-OS Release 10.6(1)F, users can configure origin-AS validation on an eBGP router and use Origin Validation State Extended Community to signal validation state to eBGP peers under their own administration.

RPKI Example

For example, a router connects to multiple RPKI caches to download ROA objects, which are then used by BGP to validate the origin-AS numbers for the internet routing table.

RPKI Configuration

RPKI configuration is a category of network configuration that organizes commands for secure routing validation.

- Commands for connecting to RPKI Caches.
- Commands for marking incoming prefixes with RPKI validation state.
- Commands for using RPKI validation state in BGP best-path computation.
- Commands for dropping or manipulating prefixes with specific validation states using route-map.

Commands for connecting to RPKI caches

The RPKI cache connection commands are used to configure and manage RPKI cache connections within the router-bgp submode.

- Configuration is performed in the **rpki-cache** submode under **router-bgp** .
- The **rpki cache** command is used to enter the submode and specify the cache IP address.
- Various parameters such as description, shutdown, transport tcp port, vrf, refresh-interval, retry-interval, and expiry-interval can be configured for each RPKI cache.

Reference information for RPKI cache connection commands

RPKI cache configuration is performed in a dedicated submode, similar to configuring BGP peers under the default VRF. The submode is accessed using the **rpki cache** command with the cache's IP address.

- description: A description to identify the cache
- shutdown: Shutdown the cache

- transport tcp port: Transport port on which cache is listening
- vrf: VRF in which RPKI cache is reachable
- refresh-interval: Specify periodic wait time between cache poll attempts
- retry-interval: Specify wait time before retrying failed serial or reset query
- expiry-interval: Specify how long to use current data while unable to perform successful query

Note Unless transport TCP port is explicitly configured, BGP will connect to RPKI cache on RPKI-RTR port 323.

Unless explicitly configured, all intervals will be determined as suggested by the RPKI Cache in End of Data PDU.

Example: Configuring an RPKI cache connection

The following example shows how to configure an RPKI cache connection and set various parameters:

```
router bgp 100
                rpki cache 147.28.0.11
                description        A description to identify the cache
                shutdown           Shutdown the cache
                transport tcp port Transport port on which cache is listening
                vrf                Vrf in which RPKI cache is reachable
                refresh-interval   Specify periodic wait time between cache poll attempts

               retry-interval     Specify wait time before retrying failed serial or reset
 query
                expiry-interval    Specify how long to use current data while unable to
perform successful query
```

Commands for marking incoming prefixes with RPKI validation state

Commands for marking incoming prefixes with RPKI validation state are configuration options that control how BGP processes and signals RPKI validation information for received prefixes.

- These commands are configured at the address-family or neighbor address-family level.
- They enable or control validation, signaling, and acceptance of RPKI validation state for eBGP and iBGP peers.
- Default behaviors may differ for eBGP and iBGP paths, and some commands are required to enable specific signaling or acceptance features.

RPKI Validation State Command Reference

Several commands are available to configure RPKI validation state processing for BGP prefixes. These commands can be applied at different configuration levels to control validation, signaling, and acceptance behavior.

- **origin-as validate** – Enables eBGP path validation against the ROA database when configured at the address-family level. This is disabled by default.

Note This command has no bearing on iBGP paths. iBGP paths are not validated against the ROA database. The only way to mark path validation state on iBGP paths is by receiving the BGP Prefix Origin Validation State Extended Community, which is done by default without configuring any command.

- **origin-as validate signal ibgp** – Enables iBGP signaling of validity state through the BGP Prefix Origin Validation State Extended Community when configured at the address-family level.
- **origin-as validate signal ebgp** – Enables eBGP signaling of validity state through the BGP Prefix Origin Validation State Extended Community to all eBGP peers when configured at the address-family level.

  When configured at the neighbor address-family level, enables eBGP signaling of validity state through the BGP Prefix Origin Validation State Extended Community to that specific eBGP peer.
- **origin-as validate accept ebgp** – Enables acceptance of validation state from eBGP peers through the BGP Prefix Origin Validation State Extended Community when configured at the address-family level.

Commands for using RPKI validation state in BGP best-path-computation

- The **bestpath origin-as use-validity** command enables the validity states of BGP paths to affect the path's preference in the BGP bestpath process. By default, this is disabled.
- The **bestpath origin-as allow invalid** command allows all "invalid" paths to be considered for BGP bestpath computation, unless best-path origin-as validate is configured. By default, this is disabled.

Reference information for RPKI validation state commands in BGP

These commands are configured at the address-family level to control the behavior of RPKI prefix validation processing in BGP.

Commands for dropping out or manipulating prefixes with specific validation states using route-map

The route-map command allows you to drop or manipulate prefixes with specific RPKI validation states, such as not-found, invalid, or valid, by matching these states in route-map clauses.

- The **not-found** parameter matches prefixes whose origin-AS is unknown in the RPKI database.
- The **invalid** parameter matches prefixes with an invalid origin-AS in the RPKI database.
- The **valid** parameter matches prefixes with a valid origin-AS in the RPKI database.

Reference Information for Route-Map RPKI Validation State Commands

The following information provides details on how route-map commands interact with RPKI validation states and their application in BGP routing.

- This match clause is relevant for inbound route-maps only.
- For iBGP learnt paths, the incoming BGP Prefix Origin Validation State Extended Community in the update will be compared against this route-map clause.

- For eBGP learnt paths, the validation state obtained by ROA database lookup will be compared against this route-map clause.

Administrators may choose to drop prefixes marked as validation-state invalid to save system memory, as these are not considered for best-path computation in BGP.

Table 29: RPKI Validation State Parameters

Parameter	Description	Usage
not-found	Origin-AS is unknown in the RPKI database.	Used to match unknown validation state prefixes.
invalid	Origin-AS is invalid in the RPKI database.	Used to match invalid validation state prefixes.
valid	Origin-AS is valid in the RPKI database.	Used to match valid validation state prefixes.

Note TIP: Use route-map clauses to control the acceptance or rejection of prefixes based on their RPKI validation state.

Example: Dropping Invalid Prefixes Using Route-Map

The following inbound route-map configuration drops prefixes with an invalid RPKI validation state to save system memory:

```
route-map sample deny 10
               match rpki invalid
               route-map sample permit 20
```

RPKI Show Commands

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

Command	Purpose
show bgp rpki summary	Displays an overview of RPKI statistics including the number of RPKI caches.

Command	Purpose
show bgp rpki table { **ipv4** | **ipv6** } { **IP address** / **masklength** }	Displays information about the current RPKI ROA database. With no options specified, the command shows the IPv4 ROA database. With the IPv6 option (show bgp rpki table ipv6), the command shows the IPv6 ROA database. ROAs that are received from a cache that is temporarily down (due to connectivity issues, for example) are displayed with (*). These ROAs will be removed from the RPKI database if the cache session does not establish within the purge-time for that cache. If an ROA prefix-block is specified after the table show command (for example, show bgp rpki table 67.21.36.0/24 max 24), then that specific ROA entry is displayed in detail, if the ROA exists. **Note** One ROA (IP address/min-max) can have multiple origin ASs and can be sourced from multiple caches.
show bgp rpki cache { **IP address** }	Displays a summary listing of all the caches that are configured and their parameters, such as **show bgp summary** . If a cache IP address is specified with the previous command, then detailed information is shown for that cache.
show bgp { **ipv4 unicast** | **ipv6 unicast** } **origin-as validity-state** { **valid** | **invalid** | **unknown** }	Displays information about BGP. This command has new options to filter the BGP table output based on path (validation_state). Specify a validity state (valid, invalid, or unknown) with this command to filter the relevant information from the BGP table, and only the BGP paths matching that validity-state are displayed.

Examples:

RPKI Clear Commands

- The **clear bgp rpki cache *** command resets the transport sessions of all configured RPKI caches.
- This command immediately purges the RPKI database of all IPv4 and IPv6 ROAs received from all caches.

RPKI Debug and Event History Commands

- The **debug bgp rpki** command enables debugging for all RPKI-related operations except prefix-validation, including cache connectivity, protocol state-machine, and ROA database events.

- The **sh bgp event-history rpki** command displays high-level information about RPKI events.

Reference Information for RPKI Debug and Event History Commands

The following commands are used to debug and review RPKI-related events in BGP:

- **debug bgp rpki** : Enables debugging for RPKI operations (excluding prefix-validation).
- **sh bgp event-history rpki** : Shows high-level RPKI event information.

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:

SUMMARY STEPS

1. **soft-reconfiguration inbound**
2. **clear bgp ipv4** {**unicast** | **multicast** *ip-address* **soft** {**in** | **out**}
3. (Optional) **clear bgp** {**ipv4** | **ipv6** } {**unicast** | **multicast** *ip-address* **soft** {**in** | **out**}
4. **clear bgp** {**ipv4** | **ipv6**} {**unicast** | **multicast**} *ip-address* **soft** (**in** | **out**)

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**soft-reconfiguration inbound** **Example:** `switch(config-router-neighbor-af)# soft-reconfiguration inbound`	Enables soft reconfiguration to store the inbound BGP route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
Step 2	**clear bgp ipv4** {**unicast** | **multicast** *ip-address* **soft** {**in** | **out**} **Example:** `switch# clear bgp ip unicast 192.0.2.1 soft in`	This command in any mode resets the BGP session without tearing down the TCP session.
Step 3	(Optional) **clear bgp** {**ipv4** | **ipv6** } {**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.

	Command or Action	Purpose
Step 4	**clear bgp** {**ipv4** | **ipv6**} {**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.

Modify the Next-Hop Address

You can modify the next-hop address used in a route advertisement in the following ways:

- Disable 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 commands in address-family configuration mode:

Procedure

Step 1 **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.

Step 2 **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.

Configure 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.

To modify the BGP next-hop address tracking, use the following commands in address-family configuration mode:

Procedure

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.

Configure 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.

To modify the BGP next-hop address tracking, use the following commands in address-family configuration mode:

Procedure

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.

Configuring Next-Hop Resolution via Default Route

BGP next-hop resolution allows you to specify if the IP default route is used for BGP next-hop resolution.

To configure BGP next-hop resolution, use the following command in router configuration mode:

SUMMARY STEPS

1. [**no**] **nexthop suppress-default-resolution**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	[**no**] **nexthop suppress-default-resolution** **Example:** `switch(config-router)# nexthop suppress-default-resolution`	Prevents resolution of BGP next hop through the IP default route. When this command is enabled: • The output of the **show bgp process detail** command includes the following line: Use default route for nexthop resolution: No • The output of the **show routing clients bgp** command includes the following line: Owned rnh will never resolve to 0.0.0.0/0

Control 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.

Shrink 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
- Line card failures
- BFD failure detections for BGP neighbors
- Administrative shutdown of BGP neighbors (using the shutdown command)

The accelerated handling of the first two events (Layer 3 link failures and line card 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 router configuration mode:

Procedure

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 routes.

Disable 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:

Procedure

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.

Disable Policy Batching

In BGP deployments where prefixes have unique attributes, BGP tries to identify routes with similar attributes to bundle in the same BGP update message. To avoid the overhead of this additional BGP processing, you can disable batching.

Cisco recommends that you disable policy batching for BGP deployments that have a large number of routes with unique next hops.

To disable policy batching, use the following command in router configuration mode:

Procedure

disable-policy-batching

Example:

```
switch(config-router)#
disable-policy-batching
```

Disables the batching evaluation of prefix advertisements to all peers.

Configure BGP Additional Paths

BGP supports sending and receiving multiple paths per prefix and advertising such paths.

Advertise 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

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# <userinput>configure terminal</userinput>
switch(config)# <userinput>router bgp 100</userinput>
switch(config-router)# <userinput>neighbor 10.131.31.2 remote-as 100</userinput>
switch(config-router-neighbor)# <userinput>address-family ipv4 unicast</userinput>
switch(config-router-neighbor-af)# <userinput>capability additional-paths send</userinput>
switch(config-router-neighbor-af)# <userinput>capability additional-paths receive</userinput>
```

Configure 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

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.

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# <userinput>configure terminal</userinput>
switch(config)# <userinput>router bgp 100</userinput>
switch(config-router)# <userinput>address-family ipv4 unicast</userinput>
switch(config-router-af)# <userinput>additional-paths send</userinput>
switch(config-router-af)# <userinput>additional-paths receive</userinput>
```

Configure 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

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 | backup | best2 | multipaths}** | advertise

Example:

```
switch(config-route-map)# set
path-selection all advertise
```

Specifies that all paths be advertised for a given prefix. You can use one of the following options:

- all—Advertises all available valid paths.
- backup—Advertises paths marked as backup paths. This option requires that backup paths be enabled using the additional-path install backup command.
- best2—Advertises the second best path, which is the best path of the remaining available paths, except the already calculated best path.
- multipaths—Advertises all multipaths. This option requires that multipaths be enabled using the maximum-paths command.



Note If there are no multipaths, the backup and best2 options are the same. If there are multipaths, best2 is the first path on the list of multipaths while backup is the best path of all available paths, except the calculated best path and multipaths.

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:

```
switch(config-route-map)# show bgp ipv4
unicast
```

Displays the path ID for the additional paths of a prefix and advertisement information for these paths.

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
```

Configure 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

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# <userinput>configure terminal</userinput>
switch(config)# <userinput>router bgp 100</userinput>
switch(config-router)# <userinput>address-family ipv4 unicast</userinput>
switch(config-router-af)# <userinput>additional-paths selection route-map
PATH_SELECTION_RMAP</userinput>
```

Configuring eBGP

Disable 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:

Procedure

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.

Configure TTL Security Hops

Perform this task to allow BGP to establish or maintain a session only if the TTL value in the IP packet header is equal to or greater than the TTL value configured for the BGP neighbor session.

Before you begin

To maximize the effectiveness of the BGP Support for TTL Security Check feature, we recommend that you configure it on each participating router. Enabling this feature secures the eBGP session in the incoming direction only and has no effect on outgoing IP packets or the remote router.

- The **neighbor ebgp-multihop** command is not needed when the BGP Support for TTL Security Check feature is configured for a multihop neighbor session and should be disabled before configuring this feature.
- The effectiveness of the BGP Support for TTL Security Check feature is reduced in large-diameter multihop peerings. In the event of a CPU utilization-based attack against a BGP router that is configured for large-diameter peering, you may still need to shut down the affected neighbor sessions to handle the attack.
- This feature is not effective against attacks from a peer that has been compromised inside of the local and remote network. This restriction also includes peers that are on the network segment between the local and remote network.

Procedure

Step 1 **enable**

Example:

```
switch(config)# enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **trace** *[protocol] destination*

Example:

```
switch(config)# trace ip 10.1.1.1
```

Discovers the routes of the specified protocol that packets will actually take when traveling to their destination.

Enter the trace command to determine the number of hops to the specified peer.

Step 3 **configure terminal**

Example:

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

Enters global configuration mode.

Step 4 **router bgp** *autonomous-system-number*

Example:

```
switch(config)# router bgp 65000
```

Enters router configuration mode, and creates a BGP routing process.

Step 5 **neighbor** *ip-address*

Example:

```
switch(config)# neighbor 10.1.1.1
```

Configures the neighbor IP address.

Step 6 **ttl-security hops** *hop-count*

Example:

```
switch(config)# ttl-security hops 2
```

Configures the maximum number of hops that separate two peers.

The hop-count argument is set to the number of hops that separate the local and remote peer. If the expected TTL value in the IP packet header is 254, then the number 1 should be configured for the hop-count argument. The range of values is a number from 1 to 254.

When the BGP Support for TTL Security Check feature is enabled, BGP will accept incoming IP packets with a TTL value that is equal to or greater than the expected TTL value. Packets that are not accepted are discarded.

The example configuration sets the expected incoming TTL value to at least 253, which is 255 minus the TTL value of 2, and this is the minimum TTL value expected from the BGP peer. The local router will accept the peering session from the 10.1.1.1 neighbor only if it is one or two hops away.

Step 7 **end**

Example:

```
switch(config)# end
```

Exits router configuration mode and enters privileged EXEC mode.

Step 8 **show running-config**

Example:

```
switch(config)# show running-config | begin bgp
```

(Optional) Displays the contents of the currently running configuration file.

The output of this command displays the configuration of the neighbor ttl-security command for each peer under the BGP configuration section of output. That section includes the neighbor address and the configured hop count.



Note Only the syntax applicable to this task is used in this example. For more details, see the Cisco IOS IP Routing: BGP Command Reference.

Step 9 **show ip bgp neighbors** *[ip-address]*

Example:

```
switch(config)# show ip bgp neighbors 10.4.9.5
```

(Optional) Displays information about the TCP and BGP connections to neighbors.

This command displays "External BGP neighbor may be up to number hops away" when the BGP Support for TTL Security Check feature is enabled. The number value represents the hop count. It is a number from 1 to 254.



Note Only the syntax applicable to this task is used in this example. For more details, see the Cisco IOS IP Routing: BGP Command Reference.

Configure 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.



Note This configuration is not supported for BGP interface peering.

To configure eBGP multihop, use the following command in neighbor configuration mode:

Procedure

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.

Disable a Fast External Fallover

By default, the Cisco Nexus 7000 Series device supports fast external fallover for neighbors in all VRFs and address-families (IPv4 or IPv6).

Be default, the Cisco NX-OS device supports fast external fallover for neighbors in all VRFs and address families (IPv4 or IPv6). Typically, when a BGP router loses connectivity to a directly connected eBGP peer, BGP triggers a fast external fallover by resetting the eBGP session to the peer. You can disable this fast external fallover to limit the instability caused by link flaps.

To disable fast external fallover, use the following command in router configuration mode:

Procedure

no fast-external-fallover

Example:

```
switch(config-router)# no
fast-external-fallover
```

Disables a fast external fallover for eBGP peers. This command is enabled by default.

Limit 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:

Procedure

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 512.

Configure 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.

This feature can only be used for true eBGP peers. You cannot use this feature for two peers that are members of different confederation subautonomous systems.

Furthermore, the remote peer's ASN configured with the remote-as command cannot be identical to the local device's ASN configured with the local-as command.

To configure eBGP local AS support, use the following command in neighbor configuration mode:

Procedure

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 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.

Example

This example shows how to configure local AS support on a VRF:

```
switch# configure terminal
 switch(config)# router bgp 1
 switch(config-router)# vrf test
 switch(config-router-vrf)# local-as 1
 switch(config-router-vrf)# show running-config bgp
```

Configure AS Confederations

To configure an AS confederation, you must specify a confederation identifier. To the outside world, the group of autonomous systems within the AS confederation look 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:

Procedure

Step 1 **confederation identifier** *as-number*

Example:

```
switch(config-router)# confederation
identifier 4000
```

In router configuration mode, this command configures a BGP confederation identifier.

The command triggers an automatic notification and session reset for the BGP neighbor sessions.

Step 2 **bgp confederation peers** *as-number* [*as-number2...*]

Example:

```
switch(config-router)# bgp confederation
peers 5 33 44
```

In router configuration mode, this command configures the autonomous systems that belong to the AS confederation.

The command specifies a list of autonomous systems that belong to the confederation and it triggers an automatic notification and session reset for the BGP neighbor sessions.

Configure 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

You must enable BGP.

Procedure

Step 1 **configure terminal**

Example:

```
switch# configure terminal
```

Enters global configuration mode.

Step 2 **router bgp***as-number*

Example:

```
switch(config)# router bgp 65535
switch(config-router)#
```

Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

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-af)# 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 65535
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 unicast IPv4 address family.

Step 9 **route-reflector-client**

Example:

```
switch(config-router-neighbor-af)#
route-reflector-client
```

Configures the device 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** {**ipv4** | **ipv6**} {**unicast** | **multicast**} **neighbors**

Example:

```
switch(config-router-neighbor-af)#
show bgp ipv4 unicast neighbors
```

Displays the BGP peers.

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)# <userinput>router bgp 65536</userinput>
switch(config-router)# <userinput>neighbor 192.0.2.10 remote-as 65536</userinput>
switch(config-router-neighbor)# <userinput>address-family ip unicast</userinput>
switch(config-router-neighbor-af)# <userinput>route-reflector-client</userinput>
switch(config-router-neighbor-af)# <userinput>copy running-config startup-config</userinput>
```

Configure Next-Hops on Reflected Routes Using an Outbound Route-Map

You can change the next-hop on reflected routes on a BGP route reflector using an outbound route-map. You can configure the outbound route-map to specify the peer's local address as the next-hop address.

Note The **next-hop-self** command does not enable this functionality for routes being reflected to clients by a route reflector. This functionality can only be enabled using an outbound route-map.

Before you begin

You must enable BGP (see the Enabling BGP section).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

You must enter the **set next-hop** command to configure an address family-specific next-hop address. For example, for the IPv6 address family, you must enter the **set ipv6 next-hop peer-address** command.

- When setting IPv4 next-hops using route-maps—If **set ip next-hop peer-address** matches the route-map, the next-hop is set to the peer's local address. If no next-hop is set in the route-map, the next-hop is set to the one stored in the path.

- When setting IPv6 next-hops using route-maps—If **set ipv6 next-hop peer-address** matches the route-map, the next-hop is set as follows:
  - For IPv6 peers, the next-hop is set to the peer's local IPv6 address.
  - For IPv4 peers, if **update-source** is configured, the next-hop is set to the source interface's IPv6 address, if any. If no IPv6 address is configured, no next-hop is set
  - For IPv4 peers, if **update-source** is not configured, the next-hop is set to the outgoing interface's IPv6 address, if any. If no IPv6 address is configured, no next-hop is set.

Procedure

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 200
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
192.0.2.12 remote-as 200
switch(config-router-neighbor)#
```

Configures the IP address and AS number for a remote BGP peer.

Step 4 (Optional) **update-source** *interface number*

Example:

```
switch(config-router-neighbor)#
update-source loopback 300
```

Specifies and updates the source of the BGP session.

Step 5 **address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**}

Example:

```
switch(config-router-neighbor)#
address-family ipv4 unicast
switch(config-router-neighbor-af)#
```

Enters router address family configuration mode for the specified address family.

Step 6 **route-reflector-client**

Example:

```
switch(config-router-neighbor-af)#
route-reflector-client
```

Configures the device 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 7 **route-map** *map-name* **out**

Example:

```
switch(config-router-neighbor-af)#
route-map setrrnh out
```

Applies the configured BGP policy to outgoing routes.

Step 8 (Optional) **show bgp** {**ipv4** | **ipv6**} {**unicast** | **multicast**} [**ip-address** | **ipv6-prefix**] **route-map** *map-name* [**vrf** *vrf-name*]

Example:

```
switch(config-router-neighbor-af)#
show bgp ipv4 unicast route-map
setrrnh
```

Displays the BGP routes that match the route map.

Step 9 (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 next-hop on reflected routes on a BGP route reflector using an outbound route-map:

```
switch(config)# <userinput>interface loopback 300</userinput>
 switch(config-if)# <userinput>ip address 192.0.2.11/32</userinput>
 switch(config-if)# <userinput>ipv6 address 2001::a0c:1a65/64</userinput>
 switch(config-if)# <userinput>ip router ospf 1 area 0.0.0.0</userinput>
 switch(config-if)# <userinput>exit</userinput>
 switch(config)# <userinput>route-map setrrnh permit 10</userinput>
 switch(config-route-map)# <userinput>set ip next-hop peer-address</userinput>
 switch(config-route-map)# <userinput>exit</userinput>
 switch(config)# <userinput>route-map setrrnhv6 permit 10</userinput>
 switch(config-route-map)# <userinput>set ipv6 next-hop peer-address</userinput>
 switch(config-route-map)# <userinput>exit</userinput>
 switch(config)# <userinput>router bgp 200</userinput>
 switch(config-router)# <userinput>neighbor 192.0.2.12 remote-as 200</userinput>
 switch(config-router-neighbor)# <userinput>update-source loopback 300</userinput>
 switch(config-router-neighbor)# <userinput>address-family ipv4 unicast</userinput>
 switch(config-router-neighbor-af)# <userinput>route-reflector-client</userinput>
 switch(config-router-neighbor-af)# <userinput>route-map setrrnh out</userinput>
 switch(config-router-neighbor-af)# <userinput>exit</userinput>
 switch(config-router-neighbor)# <userinput>address-family ipv6 unicast</userinput>
 switch(config-router-neighbor-af)# <userinput>route-reflector-client</userinput>
 switch(config-router-neighbor-af)# <userinput>route-map setrrnhv6 out</userinput>
```

Configure Route Dampening

You can configure route dampening to minimize route flaps propagating through your iBGP network.

To configure route dampening, use the following command in address-family or VRF address family configuration mode:

Procedure

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.
- *resuse-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.

Configure Load Sharing and ECMP

You can configure the maximum number of paths that BGP adds to the route table for equal-cost multipath (ECMP) load balancing.

To configure the maximum number of paths, use the following command in router address-family configuration mode:

Procedure

maximum-paths [**ibgp**] *maxpaths*

Example:

```
switch(config-router-af)# maximum-paths 8
```

Configures the maximum number of equal-cost paths for load sharing. The default is 1.

Unequal Cost Multipath (UCMP) over BGP

UCMP is also known as Weighted ECMP. It is a mechanism that allows multiple routes to the same destination with different weights per next-hop and load-balances the routed traffic over those multiple next-hops. The

basic UCMP works for most of the customers' requirements. The load entropy is the best way to maximize the link usage efficiency.

Often, the application distribution in the network can be unbalanced. The new clusters roll in at different over-subscription rates than the old clusters. The new clusters have powerful servers than the old clusters and they are capable of handling more load per CPU. As the network is not perfect, some control over routing behavior is needed. You can configure Weighted ECMP over BGP for balancing the traffic load and for administering control over the routing behavior.



Note The Link-Bandwidth Extended Community must be advertised across eBGP sessions, although it is defined as a non-transitive attribute.

Next-hop-self must strip the Link-Bandwidth Extended Community from advertisements.

Enable UCMP over BGP

The solution for the unequal distribution of the resources and sub-optimal traffic distribution use-cases is to configure Weighted ECMP over BGP. You can inject the routes (from the host or the controller) and signal a weight for each instance. You can then aggregate the weights across the infrastructure and deliver the traffic in the direct proportion to the application deployment distribution.

Guidelines and Limitations for UCMP over BGP

- BGP uses the Link-Bandwidth Extended Community defined in the draft-ietf-idr-link-bandwidth-06.txt to implement the weighted ECMP feature. The Link-Bandwidth Extended Community is advertised across eBGP sessions, although it's defined as a non-transitive attribute, as long as next-hop is unchanged.
- You can accept Link-Bandwidth Extended Community from both iBGP and eBGP peers.
- For weights programming, the Link-Bandwidth Extended Community has the link bandwidth encoded in bytes/second, as a four byte floating point integer, that is normalized between 0 and 1000 before downloading to RIB.
- The hardware ECMP width is fixed as 64 in size.

Configure 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:

Procedure

maximum-prefix *maximum* [*threshold*] [**restart** *time* | **warning-only** | **discard**]

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.
- *discard*—Introduced in NX-OS Release 10.6(2)F. Discards prefixes after the threshold is reached.

This command triggers an automatic notification and session reset for the BGP neighbor sessions if the prefix is exceeded.

Configure DSCP

You can configure a differentiated services code point (DSCP) for a neighbor. You can specify a DSCP value for locally originated packets for IPv4 or IPv6.

To configure the DSCP value, use the following command in neighbor configuration mode:

Procedure

dscp *dscp_value*

Example:

```
switch(config-router-neighbor)# dscp
                        63
```

Below is an example of the corresponding **show** command:

```
show ipv6 bgp neighbors
                        BGP neighbor is 10.1.1.1, remote AS 0, unknown link, Peer index 4
                        BGP version 4, remote router ID 0.0.0.0
                        BGP state = Idle, down for 00:13:34, retry in 0.000000
                        DSCP (DiffServ CodePoint): 0
                        Last read never, hold time = 180, keepalive interval is 60 seconds
```

Sets the differentiated services code point (DSCP) value for the neighbor. The DSCP value can be a number from 0 to 63, or it can be one of the following keywords: **ef**, **af11**, **af12**, **af13**, **af21**, **af22**, **af23**, **af31**, **af32**, **af33**, **af41**, **af42**, **af43**, **cs1**, **cs2**, **cs3**, **cs4**, **cs5**, **cs6**, or **cs7**.

The default value is cs6.

Configure Dynamic Capability

You can configure dynamic capability for a BGP peer.

To configure dynamic capability, use the following command in neighbor configuration mode:

Procedure

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.

Configure 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:

Procedure

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 filter more specific routes. If you specify the **suppress-map** option while performing a BGP route aggregation, you can set the community attribute for a BGP route update. This option enables you to set community attributes on the more-specific routes.
- The **suppress-map** keyword and argument conditionally filter more specific routes. If you specify the **suppress-map** option while performing a BGP route aggregation, you can either suppress certain more-specific routes from being advertised to its peers, or decide to advertise the more-specific routes with some community attributes set on them, depending upon the suppress-map route-map configuration. A route-map configured with only match clauses will suppress the more-specific routes that satisfy the match criteria. However, if a route-map is configured with match

and set clauses, then the routes satisfying the match criteria will be advertised with the appropriate attributes as modified by the route-map. The second option enables you to set community attributes on the more-specific routes.

Suppress 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.

To suppress BGP routes, use the following command in router configuration mode:

Procedure

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.

Configure 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.
- Nexus does not support any other BGP Attribute change operation (example prepend AS Path) with Conditional Route Advertisements. It is used to control which routes are advertised based on exist/non-exist map configuration.

If the route does not pass the condition, BGP withdraws the route if it exists in the BGP table.

Before you begin

You must enable BGP(see the Enabling BGP section).

Procedure

Step 1 **configure terminal**

Example:

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

Enters configuration mode.

Step 2 **router bgp** *as-number*

Example:

```
switch(config)# router bgp 65535
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 65534
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 advertises 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 advertises the route. The *nonexist-rmap* is a case-sensitive, alphanumeric string up to 63 characters.

Note For BGP conditional advertisement feature, ensure that the "le" or "ge" statements are not used on prefix-list when associated to exist or nonexist map.

Step 6 (Optional) **show bgp** {**ipv4** | **ipv6**} {**unicast** | **multicast**} **neighbors**

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# <userinput>configure terminal</userinput>
switch(config)# <userinput>router bgp 65536</userinput>
switch(config-router)# <userinput>neighbor 192.0.2.2 remote-as 65537</userinput>
switch(config-router-neighbor)# <userinput>address-family ipv4 unicast</userinput>
switch(config-router-neighbor-af)# <userinput>advertise-map advertise exist-map
exist</userinput>
switch(config-router-neighbor-af)# <userinput>exit</userinput>
switch(config-router-neighbor)# <userinput>exit</userinput>
switch(config-router)# <userinput>exit</userinput>
switch(config)# <userinput>route-map advertise</userinput>
switch(config-route-map)# <userinput>match as-path pathList</userinput>
switch(config-route-map)# <userinput>exit</userinput>
switch(config)# <userinput>route-map exit</userinput>
switch(config-route-map)# <userinput>match ip address prefix-list plist</userinput>
switch(config-route-map)# <userinput>exit</userinput>
switch(config)# <userinput>ip prefix-list plist permit 209.165.201.0/27</userinput>
```

Configure 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

- You must enable BGP (see the Enabling BGP section).
- Ensure that you are in the correct VDC (or use the **switchto vdc** command).

You must enable BGP.

Procedure

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 65535
switch(config-router)#
```

Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

Step 3 **address-family ipv4** {**unicast** | **multicast**}

Example:

```
switch(config-router)# address-family
ipv4 unicast
switch(config-router-af)#
```

Enters address family configuration mode.

Step 4 **address-family** {**ipv4** | **ipv6** } {**unicast** | **multicast**}

Example:

```
switch(config-router)# address-family
vpnv4 unicast
switch(config-router-af)#
```

Enters address family configuration mode.

Step 5 **address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**}

Example:

```
switch(config-router)# address-family
ipv4 unicast
switch(config-router-af)#
```

Enters address-family configuration mode.

Step 6 **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.

Step 7 **redistribute** {**direct** | {**eigrp** | **isis** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static** | **icmpv6**} **route-map** *map-name*

Example:

```
switch(config-router-af)# redistribute
eigrp 201 route-map Eigrpmap
```

Redistributes routes from other protocols into BGP.

Beginning with Cisco NX-OS Release 10.3(3)F, the keyword **icmpv6** is supported to redistribute icmpv6 routes from other protocols into BGP.

Step 8 (Optional) **default-metric** *value*

Example:

```
switch(config-router-af)# default-metric
33
```

Generates a default route into BGP.

Step 9 (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
```

DMZ Link Bandwidth

The DMZ Link Bandwidth feature is used to enable traffic load balancing towards a BGP learnt route reachable via multiple autonomous system exit links. The load balancing is done proportional to the bandwidth of these links.

Link Bandwidth Extended Community is used to carry the bandwidth of the link between two directly connected (single hop) eBGP peers. A nexus device will attach this extended community to BGP routes received from a directly connected eBGP neighbor if the dmz-link-bandwidth command is configured under that neighbor's address-family mode. This extended community is then propagated to iBGP peers when extended community exchange is enabled with the send-community extended or send-community both command. This attribute is used as a load sharing value relative to other paths in forwarding.

In addition, user may want to forcefully change the Link Bandwidth Extended Communtiy for routes received from a BGP peer. They may also only want to set this extended community for only a subset of routes received from the peer. They can achieve that by configuring an inbound route-map towards the peer and configure **set extcommunity bandwidth** *1-4000000* under it.

Guidelines and Limitations

BGP DMZ Link Bandwidth

Consider the following guidelines and limitations before configuring the Link Bandwidth feature:

- The **dmz-link-bandwidth** command can be configured only under IPv4 unicast and IPv6 unicast address families under a BGP neighbor.
- It will only attach the Link Bandwidth Extended Community to routes received from directly connected BGP neighbors. It will not do so for BGP multi-hop neighbors.
- It can be configured under both global mode and VRF mode.
- BGP multipath load balancing must be configured under the address-family using **maximum-paths** command for this feature to be enabled.
- BGP extended community exchange must be enabled between iBGP neighbors to which the Link Bandwidth Extended Community is to be advertised.
- Link Bandwidth Extended Community will be seamlessly carried to routes leaked from one VRF to another VRF.

Configure BGP DMZ Link Bandwidth

Beginning with Cisco NX-OS Release 10.5(1)F, you can configure this feature.

Procedure

Step 1 **configure terminal**

Example:

```
switch# configure terminal
```

Enters global configuration mode.

Step 2 **router bgp***as-number*

Example:

```
switch# configure terminal
```

Enters router configuration mode to create or configure a BGP routing process.

Step 3 **address-family [ipv4|ipv6] unicast**

Example:

```
switch(config)# address-family ipv4 unicast
```

Configures address family IPv4 or IPv6 unicast.

Step 4 **maximum-paths***max-path*

Example:

```
switch(config)# maximum-paths 10
```

Enable BGP multi-path under address-family.

Step 5 **template peer***peer-template-name*

Example:

```
switch(config)# template peer host_peer
```

Enters template mode and configures peer parameter.

Step 6 **address-family [ipv4 | ipv6] unicast**

Example:

```
switch(config)# address-family ipv4 unicast
```

Configures the address family for IPv4 or IPv6.

Step 7 **dmz-link-bandwidth**

Example:

```
switch(config)# dmz-link-bandwidth
```

Configures BGP to consider load balancing some traffic towards this directly connected peer by attaching link bandwidth extended community to routes received from it.

Step 8 **neighbor***neighbor*

Example:

```
switch(config)# neighbor 1.1.1.1

                      or

                      switch(config)# neighbor 11::1
```

Configure BGP neighbor.

Step 9 **remote-as***remote-as*

Example:

```
switch(config)# remote-as 100
```

Specify Autonomous System Number of the neighbor .

Step 10 **address-family [ipv4 | ipv6] unicast**

Example:

```
switch(config)# address-family ipv4 unicast
```

Configures the address family IPv4 or IPv6 unicast.

Step 11 **dmz-link-bandwidth**

Example:

```
switch(config)# dmz-link-bandwidth
```

Configures BGP to consider load balancing some traffic towards this directly connected peer by attaching link bandwidth extended community to routes received from it.

Step 12 **route-map***name* **permit***route*

Example:

```
switch(config)# route-map change_link_bandwidth permit 10
```

Confgure route-map.

Step 13 **set extcommunity bandwidth <1-4000000>**

Example:

```
switch(config-route-map)# set extcommunity bandwidth 1000
```

Configure route-map to set link bandwidth extended community.

Step 14 **neighbor***neighbor* **address-family [ipv4 | ipv6] unicast route-map***name* **in**

Example:

```
switch(config-router)# neighbor 1.1.1.1

                        switch (config-router-neighbor)# address-family ipv4 unicast

                        switch(config-router-neighbor-af)# route-map change_link_bandwidth in
```

Configure neighbor to be attached to an inbound route-map.

Configuration Examples for BGP DMZ Link Bandwidth

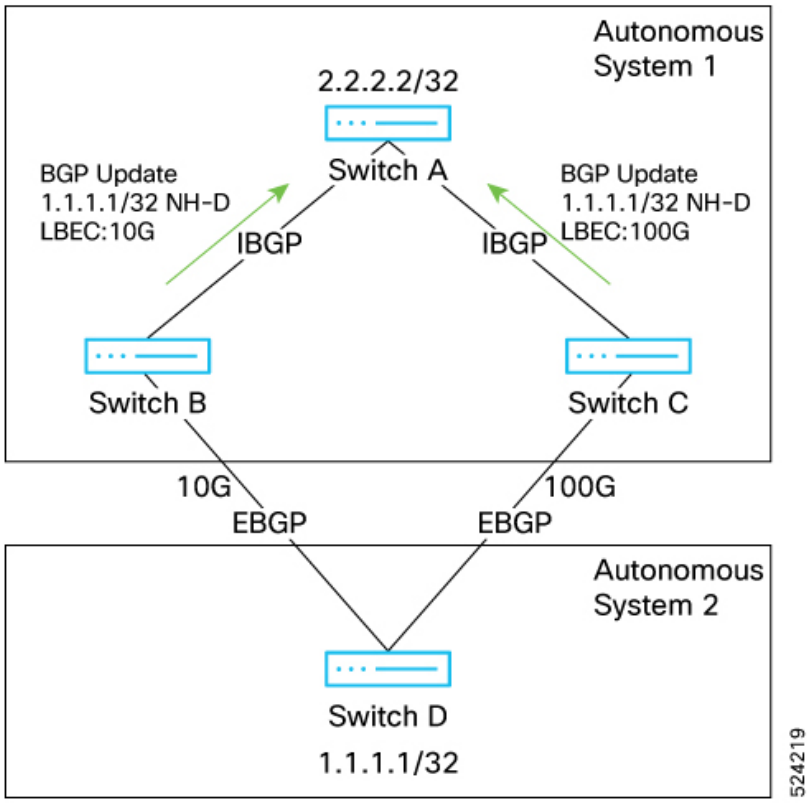
In the following example, AS 1 is attached to AS 2 through two unequal bandwidth links. B-D link is 10G and C-D link is 100G. B-D link is 10G and C-D link is 100G. A-B and A-C connected by IBGP session. B-D and C-D connected by EBGP Session.

If you want traffic to be proportionally load balanced according to these link bandwidths that is A should send 10 times the outgoing traffic to C as compared to B, configure dmz-link-bandwidth command on B and C towards EBGP neighbor D. B will package B-D bandwidth into Link Bandwidth Extended Community (LBEC) and attach it to the BGP path for route entry 1.1.1.1/32. Similarly, C will package C-D bandwidth into LBEC and attach it to the BGP path for route entry 1.1.1.1/32.

B and C will advertise the route to iBGP peer A along with LBEC.

On A, BGP will program forwarding with hash 10/110 for NH-B and 100/110 for NH-C.

Figure 33: DMZ Link Bandwidth Configuration



Switch B to D and B to A Configuration

```
router bgp 1
neighbor D
address-family ipv4|v6 unicast
dmz-link-bandwidth
neighbor A
address-family ipv4|v6 unicast
send-community extended
```

Switch C to D and C to A Configuration

```
router bgp 1
neighbor D
address-family ipv4|v6 unicast
dmz-link-bandwidth
neighbor A
address-family ipv4|v6 unicast
send-community extended
```

Switch A Control Plane and Data Plane State

BGP table state:

```
1.1.1.1/32
NH-B    LBEC: 10G
NH-C    LBEC: 100G
```

Forwarding state:

```
1.1.1.1/32
NH-B   hash 10:110
NH-C   hash 100:110
```

Configure Unequal Cost Multipath (UCMP) Using Link Bandwidth Extended Community

Before you begin

See Configuring BGP DMZ Link Bandwidth. That feature must be first configured at edge devices for this feature to work. This means that at the edge devices, the Link Bandwidth Extended Community must be attached to a BGP route received from a directly connected ebgp peer by either configuring **dmz-link-bandwidth** command or by configuring an inbound route-map with **set extcommunity link-bandwidth <1-4000000>** command. Until that happens, none of the functionality described in this section will work.

Beginning with Cisco NX-OS Release 10.5(1)F, a BGP speaker has the ability to convey to its directly connected BGP neighbor the cumulative bandwidth available from itself to a BGP learnt route if the command 'link-bandwidth cumulative' is configured under the neighbor's address-family mode. It does that by leveraging the Link Bandwidth Extended Community. It will advertise the BGP route with the value n inserted in this extended community, where n = min (Sum of bandwidth obtained from Link Bandwidth Extended Community of all available multi-paths, Bandwidth of link towards the neighbor to which advertisement is being sent).

Guidelines and Limitations

Consider the following guidelines and limitations before configuring the BGP UCMP feature:

- The **link-bandwidth cumulative** command can be configured only under IPv4 unicast and IPv6 unicast address families under a BGP neighbor.
- It will only take effect to towards directly connected BGP neighbors.
- This command will only take effect if all of the available multi-paths have the Link Bandwidth Extended Community.
- It can be configured under both global mode and vrf mode.
- Link Bandwidth Extended Community will be seamlessly carried to routes leaked from one VRF to another VRF.

Procedure

Step 1 **configure terminal**

Example:

```
switch# configure terminal
```

Enters global configuration mode.

Step 2 **router bgp** *as-number*

Example:

```
switch(config)# router bgp 120
```

Enters router configuration mode to create or configure a BGP routing process.

Step 3 **neighbor** *neighbor*

Example:

```
switch(config)# neighbor 1.1.1.1
                       or
                       switch(config)# neighbor 11::1
```

Configure BGP neighbor.

Step 4 **remote-as** *remote-as*

Example:

```
switch(config)# remote-as 100
```

Specify Autonomous System Number of the neighbor.

Step 5 **address-family [ipv4 | ipv6] unicast**

Example:

```
switch(config)# address-family ipv4 unicast
```

Configures the address family IPv4 or IPv6 unicast.

Step 6 **send-community extended**

Example:

```
switch(config)# send-community extended
```

Configures sending of BGP extended community towards this neighbor.

Step 7 **link-bandwidth cumulative**

Example:

```
switch(config)# link-bandwidth cumulative
```

Sends cumulative link bandwidth towards the neighbor. This configuration doesn't advertise the cumulative link bandwidth to the neighbor if any of the multi-paths doesn't have the link bandwidth extended community.

Configuration Example

In the following figures:

- Single hop directly connected ebgp devices in 4 clos layers.
- All links are 100G, except B-D and B-E, which are 10G. This means traffic from B to destination 1.1.1.1 is bottle necked at a 20G link bandwidth.
- User wants end-to-end traffic load balancing should account for this lower link bandwidth.

- On each device in Layer T1, configure command **dmz-link-bandwidth** towards every BGP neighbor in Layer T0.
- On each device in Layer T1, configure command **link-bandwidth cumulative** towards every BGP peer in layer T2.
- On each device in Layer T2, configure command **link-bandwidth cumulative** towards every BGP peer in layer T1 and T3.
- On each device in Layer T3, configure command **link-bandwidth cumulative** towards every BGP peer in layer T2.
- Command **dmz-link-bandwidth** will cause switch D to package the bandwidth of link D-F into Link Bandwidth Extended Community (LBEC) and attach it to the corresponding BGP path for route entry 1.1.1.1/32.
- Command **dmz-link-bandwidth** will cause switch E package the bandwidth of link E-F into LBEC and attach it to the corresponding BGP path for route entry 1.1.1.1/32.
- Command **link-bandwidth cumulative** will cause switches A, B, C, D, and E to insert bandwidth **n** into LBEC while advertising BGP update to the peers under which it is configured.
- The value **n** is calculated by using the formula min(Sum of LBEC of all multi-paths, bandwidth towards peer where update is to be advertised.)
- Propagation of LBEC in BGP update will cause forwarding on all devices to be programmed with proportional hashes
- LBEC will be dynamically recalculated in case of link failures.

Figure 34: Configuration Example 1

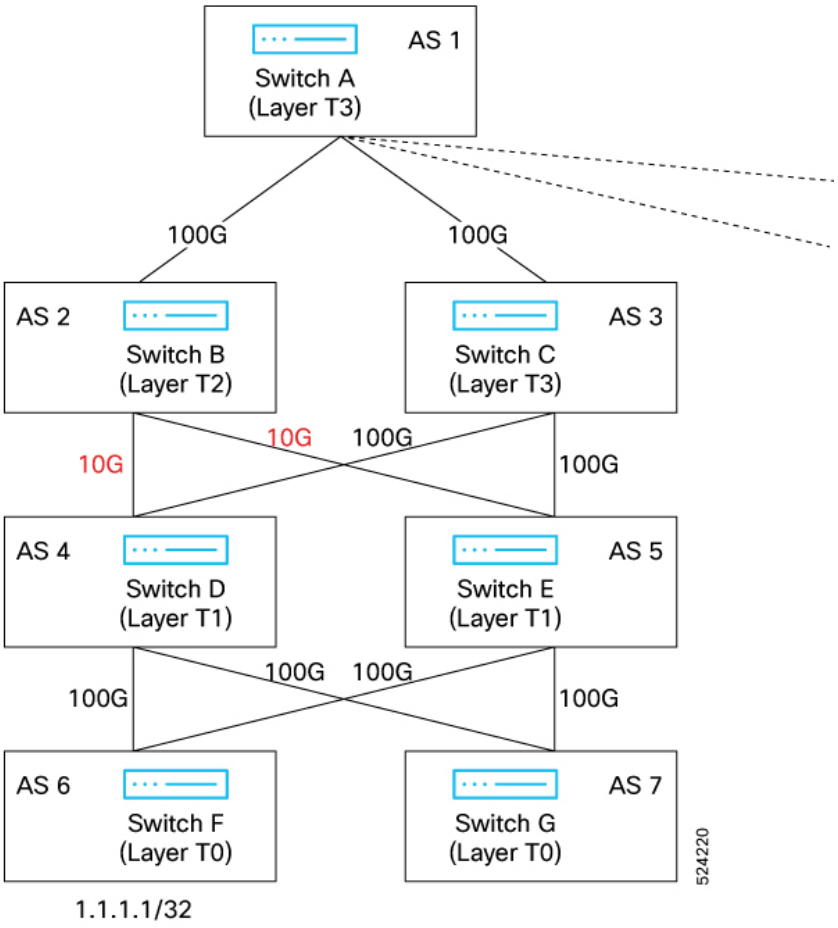
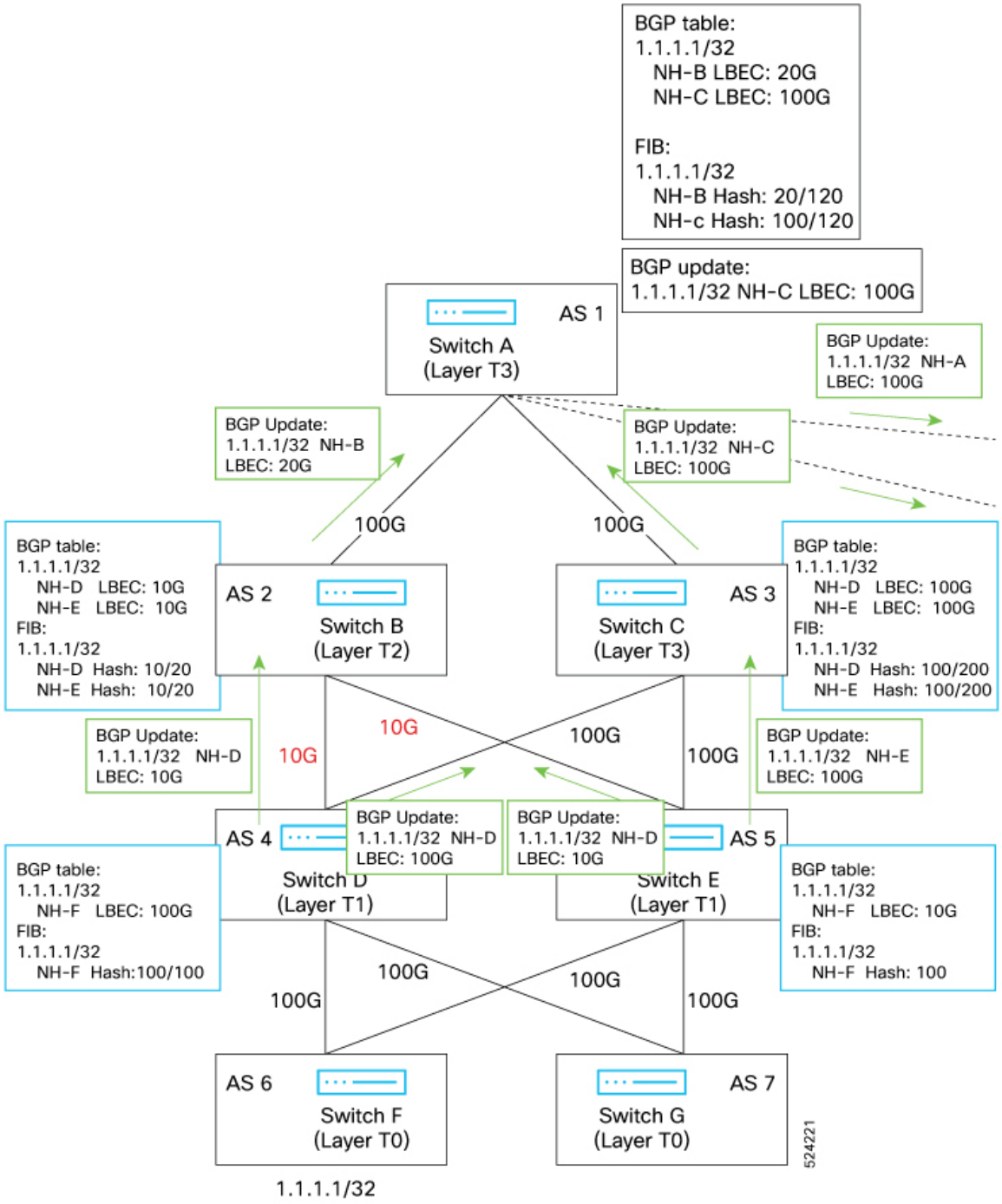


Figure 35: Configuration Example 2



Configuration Steps and Verification

To verify and configure BGP UCMP, use the following commands and configuration steps:

Configuration for A to B and C:

1.
```
router bgp 1
neighbor B
address-family ipv4|v6 unicast
link-bandwidth cumulative
send-community extended
neighbor C
address-family ipv4|v6 unicast
link-bandwidth cumulative
send-community extended
neighbor B'
address-family ipv4|v6 unicast
link-bandwidth cumulative
send-community extended
neighbor C'
address-family ipv4|v6 unicast
link-bandwidth cumulative
send-community extended
```

Configuration for B to D, A, and E:

1.
```
router bgp 1
neighbor B
address-family ipv4|v6 unicast
link-bandwidth cumulative
send-community extended
neighbor C
address-family ipv4|v6 unicast
link-bandwidth cumulative
send-community extended
neighbor B'
address-family ipv4|v6 unicast
link-bandwidth cumulative
send-community extended
neighbor C'
address-family ipv4|v6 unicast
link-bandwidth cumulative
send-community extended
```

Configuration for C to A, D, and E:

1. `router bgp 3`
```
neighbor A
address-family ipv4|v6 unicast
link-bandwidth cumulative
send-community extended
neighbor D
address-family ipv4|v6 unicast
link-bandwidth cumulative
send-community extended
neighbor E
address-family ipv4|v6 unicast
link-bandwidth cumulative
send-community extended
```

Configuration for D to F, B, and C:

1. `router bgp 4`
```
neighbor F
address-family ipv4|v6 unicast
dmz-link-bandwidth
neighbor B
```

```
                        address-family ipv4|v6 unicast
                        link-bandwidth cumulative
                        send-community extended
                        neighbor C
                        address-family ipv4|v6 unicast
                        link-bandwidth cumulative
                        send-community extended
```

Configuration for E to F, B, and C:

1. `router bgp 5`

```
                        neighbor F
                        address-family ipv4|v6 unicast
                        dmz-link-bandwidth
                        neighbor B
                        address-family ipv4|v6 unicast
                        link-bandwidth cumulative
                        send-community extended
                        neighbor C
                        address-family ipv4|v6 unicast
                        link-bandwidth cumulative
                        send-community extended
```

Verify Configuration

Use the following commands to verify configuration:

- Use the following command to see if the dmz-link-bandwidth command is enabled towards a peer

```
show bgp ipv4 unicast neighbors 192.168.11.2 | i i link
                        dmz-link-bandwidth is enabled
```

- Use the following command to see if the link-bandwidth cumulative command is enabled towards a peer

```
show bgp ipv4 unicast neighbors 10.1.1.2 | i i link
                        link-bandwidth cumulative is enabled
```

- Use the following command to confirm if BGP path has Link Bandwidth Extended Community

```
show bgp ipv4 unicast 1.1.1.1/32
                        BGP routing table information for VRF default, address family
IPv4 Unicast
                        BGP routing table entry for 1.1.1.1/32, version 403 Paths: (1
available, best #1) Flags: (0x8000001a) (high32 0x002000) on xmit-list, is in urib, is
 best urib route, is in HW
                        Advertised path-id 1 Path type: external, path is valid, is
best path, no labeled nexthop, in rib

                        AS-Path: 10 33299 51178 47751 {27016} , path sourced external
to AS 192.168.11.2 (metric 0) from 192.168.11.2 (192.168.11.2) Origin EGP, MED 2219,
localpref 100, weight 0 Community: 1:1 Extcommunity: LB:1:125000000

                        Path type: external, path is valid, is multi-path, no labeled
nexthop, in rib
                        AS-Path: 10 33299 51178 47751 {27016} , path sourced external
to AS 192.168.11.3 (metric 0) from 192.168.11.2 (192.168.11.2) Origin EGP, MED 2219,
localpref 100, weight 0 Community: 1:1 Extcommunity: LB:1:250000000
```

- Use the following command to confirm if routing table has the hashing ratios

```
 show ip route 1.1.1.1/32 detail

                        100.1.1.1/32, ubest/mbest: 1/0 *via 192.168.11.2, [20/2219],
00:14:22, bgp-1, bw:333, external, tag 10 client-specific data: 10 recursive next hop:
 192.168.11.2/32 extended route information: BGP origin AS 0 BGP peer AS 10
                        100.1.1.2/32, ubest/mbest: 1/0 *via 192.168.11.3, [20/2219],
00:14:22, bgp-1, bw:666, external, tag 10 client-specific data: 10 recursive next hop:
 192.168.11.3/32 extended route information: BGP origin AS 0 BGP peer AS 10
```

Advertise the Default Route

You can configure BGP to advertise the default route (network 0.0.0.0).

Before you begin

You must enable BGP (see the Enabling BGP section).

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **route-map allow permit**

Example:

```
switch(config)# route-map allow permit
switch(config-route-map)#
```

Enters router map configuration mode and defines the conditions for redistributing routes.

Step 3 **exit**

Example:

```
switch(config-route-map)# exit
switch(config)#
```

Exits router map configuration mode.

Step 4 **ip route** *ip-address network-mask* **null** *null-interface-number*

Example:

```
switch(config)# ip route 192.0.2.1 255.255.255.0 null 0
```

Configures the IP address.

Step 5 **router bgp** *as-number*

Example:

```
switch(config)# router bgp 65535
switch(config-router)#
```

Enters BGP mode and assigns the AS number to the local BGP speaker.

Step 6 **address-family** {**ipv4** | **ipv6**} **unicast**

Example:

```
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)#
```

Enters address-family configuration mode.

Step 7 **default-information originate**

Example:

```
switch(config-router-af)# default-information originate
```

Advertises the default route.

Step 8 **redistribute static route-map allow**

Example:

```
switch(config-router-af)# redistribute static route-map allow
```

Redistributes the default route.

Step 9 (Optional) **copy running-config startup-config**

Example:

```
switch(config-router-af)# copy running-config startup-config
```

Saves this configuration change.

Configure BGP Attribute Filtering and Error Handling

Beginning with Cisco NX-OS Release 9.3(3), you can configure BGP attribute filtering and error handling to provide an increased level of security.

The following features are available and implemented in the following order:

- **Path attribute treat-as-withdraw:** Allows you to treat-as-withdraw a BGP update from a specific neighbor if the update contains a specified attribute type. The prefixes contained in the update are removed from the routing table.
- **Path attribute discard:** Allows you to remove specific path attributes in a BGP update from a specific neighbor.
- **Enhanced attribute error handling:** Prevents peer sessions from flapping due to a malformed update.

Attribute types 1, 2, 3, 4, 5, 8, 14, 15, and 16 cannot be configured for path attribute treat-as-withdraw and path attribute discard. Attribute type 9 (Originator) and type 10 (Cluster-id) can be configured for eBGP neighbors only.

Treat as Withdraw Path Attributes from a BGP Update Message

To **treat-as-withdraw** BGP updates that contain specific path attributes, use the following command in router neighbor configuration mode:

Procedure

[**no**] **path-attribute treat-as-withdraw** [*value* | **range** *start end*] **in**

Example:

```
switch#(config-router)# neighbor 10.20.30.40
            switch(config-router-neighbor)# path-attribute treat-as-withdraw 100 in
```

Example:

```
switch#(config-router)# neighbor 10.20.30.40
            switch(config-router-neighbor)# path-attribute treat-as-withdraw range 21 255 in
```

Treats as withdraw any incoming BGP update messages that contain the specified path attribute or range of path attributes and triggers an inbound route refresh to ensure that the routing table is up to date. Any prefixes in a BGP update that are treat-as-withdraw are removed from the BGP routing table.

This command is also supported for BGP template peers and BGP template peer sessions.

Discarding Path Attributes from a BGP Update Message

To discard BGP updates that contain specific path attributes, use the following command in router neighbor configuration mode:

Procedure

	Command or Action	Purpose

Step 1	[**no**] **path-attribute discard** [*value* | **range** *start end*] **in**
	Example:
	`switch#(config-router)# neighbor 10.20.30.40 switch(config-router-neighbor)# path-attribute discard 100 in`
	Example:
	`switch#(config-router)# neighbor 10.20.30.40 switch(config-router-neighbor)# path-attribute discard range 100 255 in`
		Drops specified path attributes in BGP update messages for the specified neighbor and triggers an inbound route refresh to ensure that the routing table is up to date. You can configure a specific attribute or an entire range of unwanted attributes.
		This command is also supported for BGP template peers and BGP template peer sessions.
		Note When the same path attribute is configured higher priority.

Enable or Disable Enhanced Attribute Error Handling

BGP enhanced attribute error handling is enabled by default but can be disabled. This feature, which complies with RFC 7606, prevents peer sessions from flapping due to a malformed update. The default behavior applies to both eBGP and iBGP peers.

To disable or reenable enhanced error handling, use the following command in router configuration mode:

Procedure

[**no**] **enhanced-error**

Example:

```
switch(config)# router bgp 1000
switch(config-router)# enhanced-error
```

Enables or disables BGP enhanced attribute error handling.

Display Discarded or Unknown Path Attributes

To display information about discarded or unknown path attributes, perform one of the following tasks:

Command	Purpose
show bgp {**ipv4** | **ipv6**} **unicast path-attribute discard**]	Displays all prefixes for which an attribute has been discarded.
show bgp {**ipv4** | **ipv6**} **unicast path-attribute unknown**]	Displays all prefixes that have an unknown attribute.
show bgp {**ipv4** | **ipv6**} **unicast** *ip-address*	Displays the unknown attributes and discarded attributes associated with a prefix.

The following example shows the prefixes for which an attribute has been discarded:

```
switch# show bgp ipv4 unicast path-attribute discard
Network            Next Hop
1.1.1.1/32         20.1.1.1
1.1.1.2/32         20.1.1.1
1.1.1.3/32         20.1.1.1
```

The following example shows the prefixes that have an unknown attribute:

```
switch# show bgp ipv4 unicast path-attribute unknown
Network            Next Hop
2.2.2.2/32         20.1.1.1
2.2.2.3/32         20.1.1.1
```

The following example shows the unknown attributes and discarded attributes associated with a prefix:

```
switch# show bgp ipv4 unicast 2.2.2.2
BGP routing table entry for 2.2.2.2/32, version 6241
Paths: (1 available, best #1, table default)
  Not advertised to any peer
  Refresh Epoch 1
  1000
    20.1.1.1 from 20.1.1.1 (20.1.1.1)
      Origin IGP, localpref 100, valid, external, best
      unknown transitive attribute: flag 0xE0 type 0x62 length 0x64
        value 0000 0000 0100 0000 0200 0000 0300 0000
              0400 0000 0500 0000 0600 0000 0700 0000
```

```
        0800 0000 0900 0000 0A00 0000 0B00 0000
        0C00 0000 0D00 0000 0E00 0000 0F00 0000
        1000 0000 1100 0000 1200 0000 1300 0000
        1400 0000 1500 0000 1600 0000 1700 0000
        1800 0000
rx pathid: 0, tx pathid: 0x0
Updated on Jul 20 2019 07:50:43 PST
```

Tune BGP

You can tune BGP characteristics through a series of optional parameters.

To tune BGP, use the following optional commands in router configuration mode:

Command	Purpose
bestpath [**always-compare-med** | **as-pathmultipath-relax** | **compare-routerid** |**cost-community ignore** | **igp-metric ignore** |**med** {**confed** |**missing-as-worst**| **non-deterministic**}] **Example:** `switch(config-router)# bestpath always-compare-med`	Modifies the best-path algorithm. The optional parameters are as follows: • **always-compare-med** —Compares MED on paths from different autonomous systems. • **as-path multipath-relax** — Allows load sharing across the providers with different (but equal-length) AS paths. Without this option, the AS paths must be identical for load sharing. When configured, BGP selects the best path with the lowest MED among potential multipaths, even if they come from different ASNs. • **compare-routerid** —Compares the router IDs for identical eBGP paths. • **cost-community ignore** —Ignores the cost community for BGP best-path calculations. • **igp-metric ignore** —Ignores the Interior Gateway Protocol (IGP) metric for next hop during best-path selection. This option is supported beginning with Cisco NX-OS Release 9.2(2). • **med confed** —Forces bestpath to do a MED comparison only between paths originated within a confederation. • **med missing-as-worst** —Treats a missing MED as the highest MED. • **med non-deterministic** —Does not always pick the best MED path from among the paths from the same autonomous system.

Command	Purpose
enforce-first-as **Example:** `switch(config-router)# enforce-first-as`	Enforces the neighbor autonomous system to be the first AS number listed in the AS_path attribute for eBGP.
log-neighbor-changes **Example:** `switch(config-router)# log-neighbor-changes`	Generates a system message when any neighbor changes state. **Note** To suppress neighbor status change messages for a specific neighbor, you can use the **log-neighbor-changes disable** command in router address-family configuration mode.
router-id *id* **Example:** `switch(config-router)# router-id 10.165.20.1`	Manually configures the router ID for this BGP speaker.
timers [**bestpath-delay** *delay* | *bgpkeepalive holdtime* | **prefix-peer-timeout** *timeout* | **bestpath-limit** *bestpath-timeout*] **Example:** `switch(config-router)# timers bestpath-limit 300`	Sets BGP timer values. The optional parameters are as follows: • *delay* —Initial best-path timeout value after a restart. The range is from 0 to 3600 seconds. The default value is 300. • *keepalive* —BGP session keepalive time. The range is from 0 to 3600 seconds. The default value is 60. • *holdtime* —BGP session hold time. The range is from 0 to 3600 seconds. The default value is180. • *timeout* —Prefix peer timeout value. The range is from 0 to 1200 seconds. The default value is 30. • *bestpath-timeout* —Bestpath timeout in seconds. The default value is 300. When a high-scale BGP setup is expected, the timeout value needs to be set to 480. You must manually reset the BGP sessions after configuring this command.

To tune BGP, use the following optional commands in router address-family configuration mode:

Command	Purpose
distance *ebgp-distance ibgp-distance local-distance* **Example:** `switch(config-router-af)# distance 20 100 200`	Sets the administrative distance for BGP. The range is from 1 to 255. The defaults are as follows: • *ebgp-distance* —20. • *ibgp-distance* —200. • *local-distance* —220. Local-distance is the administrative distance used for aggregate discard routes when they are installed in the RIB. After you enter the value for the external administrative distance, you must enter the value for the administrative distance for the internal routes or/and the value for the administrative distance for the local routes depending on your requirement; so that the internal/local routes are also considered in the route administration.
log-neighbor-changes [**disable**] **Example:** `switch(config-router-af)# log-neighbor-changes disable`	Generates a system message when this specific neighbor changes state. The **disable** option suppresses neighbor status changes messages for this specific neighbor.

To tune BGP, use the following optional commands in neighbor configuration mode:

Command	Purpose
description *string* **Example:** `switch(config-router-neighbor)# description main site`	Sets a descriptive string for this BGP peer. The string can be up to 80 alphanumeric characters.
low-memory exempt **Example:** `switch(config-router-neighbor)# low-memory exempt`	Exempts this BGP neighbor from a possible shutdown due to a low memory condition.
transport connection-mode passive **Example:** `switch(config-router-neighbor)# transport connection-mode passive`	Allows a passive connection setup only. This BGP speaker does not initiate a TCP connection to a BGP peer. You must manually reset the BGP sessions after configuring this command.

Command	Purpose
[**no** | **default**] **remove-private-as** [**all** |**replace-as**] **Example:** `switch(config-router-neighbor)# remove-private-as`	Removes private AS numbers from outbound route updates to an eBGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions. The optional parameters are as follows: • **no** —Disables the command. • **default** —Moves the command to its default mode. • **all** —Removes all private-as numbers from the AS-path value. • **replace-as** —Replaces all private AS numbers with the replace-as AS-path value. See the *Guidelines and Limitations for BGP* section for additional information on this command.
update-source *interface-type number* **Example:** `switch(config-router-neighbor)# update-source ethernet 2/1`	Configures the BGP speaker to use the source IP address of the configured interface for BGP sessions to the peer. This command triggers an automatic notification and session reset for the BGP neighbor sessions. Single-hop iBGP peers support fast external fallover when **update-source** is configured.

To tune BGP, use the following optional commands in neighbor address-family configuration mode:

Command	Purpose
allowas in **Example:** `switch(config-router-neighbor-af)# allowas in`	Allows routes that have their own AS in the AS path to be installed in the BRIB.
default-originate [**route-map** *map-name*] **Example:** `switch(config-router-neighbor-af)# default-originate`	Generates a default route to the BGP peer.
disable-peer-as-check **Example:** `switch(config-router-neighbor-af)# disable-peer-as-check`	Disables peer AS-number checking while the device advertises routes learned from one node to another node in the same AS path.

Command	Purpose
filter-list *list-name* {**in** | **out**} **Example:** `switch(config-router-neighbor-af)# filter-list BGPFilter in`	Applies an AS_path filter list to this BGP peer for inbound or outbound route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
prefix-list *list-name* {**in** | **out**} **Example:** `switch(config-router-neighbor-af)# prefix-list PrefixFilter in`	Applies a prefix list to this BGP peer for inbound or outbound route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
send-community **Example:** `switch(config-router-neighbor-af)# send-community`	Sends the community attribute to this BGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
send-community extended **Example:** `switch(config-router-neighbor-af)# send-community extended`	Sends the extended community attribute to this BGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
suppress-inactive **Example:** `switch(config-router-neighbor-af)# suppress-inactive`	Advertises the best (active) routes only to the BGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
[no | default] as-override **Example:** `switch(config-router-neighbor-af)# as-override`	**no** - (Optional) Disables the command. **default** - (Optional) Moves the command to its default mode. **as-override** - While sending updates to eBGP peer, replaces in the *path* attribute all occurrences of the peer's AS number with the local AS number.

Configure Policy-Based Administrative Distance

You can configure a distance for external BGP (eBGP) and internal BGP (iBGP) routes that match a policy described in the configured route map. The distance configured in the route map is downloaded to the unicast RIB along with the matching routes. BGP uses the best path to determine the administrative distance when downloading next hops in the unicast RIB table. If there is no match or a deny clause in the policy, BGP uses the distance configured in the distance command or the default distance for routes.

The policy-based administrative distance feature is useful when there are two or more different routes to the same destination from two different routing protocols.

Before you begin

You must enable BGP.

Procedure

Step 1 switch# **configure terminal**

Enters global configuration mode.

Step 2 switch(config)# **ip prefix-list** *name* **seq** *number* **permit** *prefix-length*

Creates a prefix list to match IP packets or routes with the permit keyword.

Step 3 switch(config)# **route-map** *map-tag* **permit** *sequence-number*

Creates a route map and enters route-map configuration mode with the permit keyword. If the match criteria for the route is met in the policy, the packet is policy routed.

Step 4 switch(config-route-map)# **match ip address prefix-list** *prefix-list-name*

Matches IPv4 network routes based on a prefix list. The prefix-list name can be any alphanumeric string up to 63 characters.

Step 5 switch(config-route-map)# **set distance** *value1 value2 value3*

Specifies the administrative distance for interior BGP (iBGP) or exterior BGP (eBGP) routes and BGP routes originated in the local autonomous system. The range is from 1 to 255.

After you enter the value for the external administrative distance, you must enter the value for the administrative distance for the internal routes or/and the value for the administrative distance for the local routes depending on your requirement; so that the internal/local routes are also considered in the route administration.

Step 6 switch(config-route-map)# **exit**

Exits route-map configuration mode.

Step 7 switch(config)# **router bgp** *as-number*

Enters BGP mode and assigns the AS number to the local BGP speaker.

Step 8 switch(config-router)# **address-family** {**ipv4** | **ipv6** | **vpnv4** | **vpnv6**} **unicast**

Enters address family configuration mode.

Step 9 switch(config-router-af)# **table-map** *map-name*

Configures the selective administrative distance for a route map for BGP routes before forwarding them to the RIB table. The table-map name can be any alphanumeric string up to 63 characters.



Note You can also configure the **table-map** command under the VRF address-family configuration mode.

Step 10 (Optional) switch(config-router-af)# **show forwarding distribution**

Displays forwarding information distribution.

Step 11 (Optional) switch(config)# **copy running-config startup-config**

Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Configure Multiprotocol BGP

You can configure MP-BGP to support multiple address families, including IPv4 and IPv6 unicast and multicast routes.

Before you begin

- You must enable BGP (see the Enabling BGP section).

You must enable BGP.

Procedure

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 65535
  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 65534
  switch(config-router-neighbor)#
```

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.

Step 4 switch(config-router-neighbor)# **address-family ipv4** {**unicast** | **multicast**}

Enters address family configuration mode.

Step 5 switch(config-router-neighbor)# **address-family** {**ipv4** | **ipv6** } {**unicast** | **multicast**}

Enters address family configuration mode.

Step 6 **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 7 (Optional) **copy running-config startup-config**

Example:

```
switch(config-router-neighbor-af)# copy
  running-config startup-config
```

Saves this configuration change.

This example shows how to enable advertising and receiving IPv4 routes for multicast RPF for a neighbor:

```
switch# configure terminal
switch(config)# interface ethernet 2/1
switch(config-if)# ipv4 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 ipv4 multicast
switch(config-router-neighbor-af)# copy running-config startup-config
```

This example shows how to enable advertising and receiving IPv4 and IPv6 routes for multicast RPF for a neighbor:

```
switch# configure terminal
switch(config)# interface ethernet 2/1
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
```

Configure BMP

Beginning with Cisco NX-OS Release 7.0(3)I5(2), you can configure BMP on the device.

Before you begin

You must enable BGP (see the Enabling BGP section).

Procedure

Step 1 **configure terminal**

Example:

```
switch# configure terminal
```

Enters global configuration mode.

Step 2 **router bgp as-number**

Example:

```
switch(config)# router bgp 200
```

Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

Step 3 **bmp server** *server-number*

Example:

```
switch(config-router-bmp)# bmp-server 1
```

Configures the BMP server to which BGP should send information. The server number is used as a key.

Note You can configure up to two BMP servers.

Step 4 **address ip-address port-number port-number**

Example:

```
switch(config-router-bmp)# address 10.1.1.1 port-number 2000
```

Configures the IPv4 or IPv6 address of the host and the port number on which the BMP speaker connects to the BMP server.

Step 5 **description** *string*

Example:

```
switch(config-router-bmp)# description BMPserver1
```

Configures the BMP server description. You can enter up to 256 alphanumeric characters.

Step 6 **initial-refresh** { *skip* | *delay time*}

Example:

```
switch(config-router-bmp)# initial-refresh delay 100
```

Configures the option to send a route refresh when BGP is converged and the BMP server connection is established later.

The skip option specifies to not send a route refresh if the BMP server connection comes up later.

The delay option specifies the time in seconds after which the route refresh should be sent. The range is from 30 to 720 seconds, and the default value is 30 seconds.

Step 7 **initial-delay** *time*

Example:

```
switch(config-router-bmp)# initial-delay 120
```

Configures the delay after which a connection is attempted to the BMP server. The range is from 30 to 720 seconds, and the default value is 45 seconds.

Step 8 **stats-reporting-period** *time*

Example:

```
switch(config-router-bmp)# stats-reporting-period 50
```

Configures the time interval in which the BMP server receives the statistics report from BGP neighbors. The range is from 30 to 720 seconds, and the default is disabled.

Step 9 shutdown

Example:

```
switch(config-router-bmp)# shutdown
```

Disables the connection to the BMP server.

Step 10 **vrf** *vrf-name*

Example:

```
switch(config-router-bmp)# vrf BMP
```

Selects vrf in which BMP server is reachable.

Step 11 **update-source** <*interface-name*>

Example:

```
switch(config-router-bmp)# update-source ethernet4/2
```

Selects local interface to be used for establishing BMP server connection.

Step 12 **neighbor ip-address**

Example:

```
switch(config-router-bmp)# neighbor 192.168.1.2
```

Enters neighbor configuration mode for BGP routing and configures the neighbor IP address.

Step 13 **remote-as** *as-number*

Example:

```
switch(config-router-neighbor)# remote-as 65535
```

Configures the AS number for a remote BGP peer.

Step 14 **bmp-activate-server** *server-number*

Example:

```
switch(config-router-neighbor)# bmp-activate-server 1
```

Configures the BMP server to which a neighbor's information should be sent.

Step 15 (Optional) **show bgp bmp** *server [server-number] [detail]*

Example:

```
switch(config-router-neighbor)# show bgp bmp server
```

Displays BMP server information.

Step 16 (Optional) **copy running-config startup-config**

Example:

```
switch(config-router-neighbor)# copy running-config startup-config
```

Saves this configuration change.

BGP Local Route Leaking

BGP Local Route Leaking

Beginning with release 9.3(1), NX-OS BGP supports leaking imported VPN routes between:

- The VPN route table and default VRF route table
- The VPN route table and VRF-lite route table
- Border leaf (BL) switch route tables for leaf-to-leaf connectivity

This feature enables the propagation of routes between the route tables. You can control route leaking for a VRF by configuring an import or export map, which now includes an option to allow or prevent incoming locally originated routes and specify whether they should be advertised. Local route leaking is bidirectional, so routes that are locally originated can be leaked from a VRF out to a BGP VPN, and routes that are imported from the BGP VPN can be leaked into a VRF.

Note NX-OS supports a similar feature called centralized route leaking. For information, see Configure Layer 3 virtualization, on page 487.

Guidelines and Limitations for BGP Local Route Leaking

The following are the guidelines and limitations for the BGP local route leaking feature:

- The following Cisco hardware supports this feature:
  - Cisco Nexus 9332C, 9364C, 9300-EX, 9300-FX/FXP/FX2/FX3, and 9300-GX platform switches, and Cisco Nexus 9500 platform switches with 9700-EX/FX line cards
  - Cisco Nexus 9500 platform switches with -R line cards
- When using route-targets, the same route-targets might have duplicated paths pointing to the same remote path, which can negatively impact the switch's memory and performance. Be careful when using route targets.
- Be careful when using local route leaking in a leaf-to-leaf case, where border-leaf routers (BLs) are leaking between the same VRFs. This scenario is more prone to routing loops. We recommend using inbound route-maps to exclude the imported routes from other BLs.
- After a remote path gets withdrawn, it can take up to 20 seconds more for BGP to completely clean up the path.

Configure Routes Imported from a VPN to Leak into the Default VRF

You can configure a VRF to allow routes that are imported from a BGP VPN to be exported to the default VRF. Use this procedure for a non-default VRF.

Before you begin

If you have not already enabled BGP, enable it now (**feature bgp**).

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **vrf context** *vrf-name*

Example:

```
switch-1(config)# vrf context vpn1
                        switch-1(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 **address-family** *address-family sub family*

Example:

```
switch-1(config-vrf)# address-family ipv4 unicast
                        switch-1(config-vrf-af-ipv4)#
```

Step 4 **export vrf default [***prefix-limit***] map***route-map* **allow-vpn**

Example:

```
switch-1(config-vrf-af-ipv4)# export vrf default map vpnmap1 allow-vpn
                        switch-1(config-vrf-af-ipv4)#
```

Configures the current VRF to allow routes that are imported from a BGP VPN to be exported to the default VRF.

Configure Routes Leaked from the Default-VRF to Export to a VPN

Before you begin

If you have not already enabled BGP, enable it now (**feature bgp**).

You can configure a VRF to allow routes leaked from the default VRF to be exported to a BGP VPN. Use this procedure for a non-default VRF.

Procedure

Step 1 **config terminal**

Example:

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

Enters global configuration mode.

Step 2 **vrf context** *vrf-name*

Example:

```
switch-1(config)# vrf context vpn1
                        switch-1(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 **address-family** *address-family sub family*

Example:

```
switch-1(config-vrf)# address-family ipv4 unicast
                        switch-1(config-vrf-af-ipv4)#
```

Step 4 **import vrf default [***prefix-limt***] map***route-map* **advertise-vpn**

Example:

```
switch-1(config-vrf-af-ipv4)# import vrf map vpnmap1 advertise-vpn
                        switch-1(config-vrf-af-ipv4)#
```

Configures the current VRF to allow routes imported from the default VRF to be exported to a BGP VPN.

Configure Routes Imported from a VPN to Export to a VRF

You can configure a VRF to allow VPN imported routes to be exported to another VRF. Use this procedure for non-default VRFs.

Before you begin

If you have not already enabled BGP, enable it now (**feature bgp**).

Procedure

Step 1 **config terminal**

Example:

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

Enters global configuration mode.

Step 2 **vrf context** *vrf-name*

Example:

```
switch-1(config)# vrf context vpn1
                        switch-1(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 **address-family** *address-family sub family*

Example:

```
switch-1(config-vrf)# address-family ipv4 unicast
                      switch-1(config-vrf-af-ipv4)#
```

Step 4 **export vrf allow-vpn**

Example:

```
switch-1(config-vrf-af-ipv4)# export vrf allow-vpn
                      nxosv2(config-vrf-af-ipv4)#
```

Configures a VRF to allow routes imported from a BGP VPM to be exported to a non-default VRF.

Configure Routes Imported from a VRF to Export to a VPN

You can configure a VRF to allow routes imported from another VRF to be exported to a BGP VPN. Use this procedure for non-default VRFs.

Before you begin

If you have not already enabled BGP, enable it now (**feature bgp**).

Procedure

Step 1 **config terminal**

Example:

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

Enters global configuration mode.

Step 2 **vrf context** *vrf-name*

Example:

```
switch-1(config)# vrf context vpn1
                      switch-1(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 **address-family** *address-family sub family*

Example:

```
switch-1(config-vrf)# address-family ipv4 unicast
                      switch-1(config-vrf-af-ipv4)#
```

Step 4 **import vrf advertise-vpn**

Example:

```
switch-1(config-vrf-af-ipv4)# import vrf advertise-vpn
                        nxosv2(config-vrf-af-ipv4)#
```

Configures the current VRF to allow routes that are imported from another VRF to be exported to a BGP VPN.

Configuration Examples

The following show sample configurations for the BGP local route leaking feature.

Configure BGP VPN to Default VRF Reachability

In this example, the configuration enables route re-importation through an intermediate VRF, called VRF_A, which is between the VPN and the default VRF.

```
vrf context VRF_A
   address-family ipv4 unicast
   route-target both auto evpn
   import vrf default map MAP_1 advertise-vpn
   export vrf default map MAP_1 allow-vpn
```

Route re-importation is enabled by using the **advertise-vpn** option to control importing routes from the VPN into VRF_A, and **allow-vpn** for the export map to control exporting VPN-imported routes from VRF_A out to the default VRF. Configuration occurs on the intermediate VRF.

Configuring VPN to VRF-Lite Reachability

In this example, the VPN connects to a tenant VRF, called VRF_A. VRF_A connects a VRF-Lite, called VRF-B. The configuration enables VPN imported routes to be leaked from VRF_A to VRF_B.

```
vrf context VRF_A
   address-family ipv4 unicast
   route-target both auto
   route-target both auto evpn
   route-target import 3:3
   route-target export 2:2
   import vrf advertise-vpn
   export vrf allow-vpn
vrf context VRF_B
   address-family ipv4 unicast
   route-target both 1:1
   route-target import 2:2
   route-target export 3:3
```

Route leaking between the two is enabled by using the **allow-vpn** in an export map configured in VRF_A (tenant). The export map in VRF_A allows route imported from the VPN to be leaked into the VRF_B. Routes processed by the export map have the **route-mapexport** and **export-map** attributes added to the route's set of route targets. The import map uses **advertise-vpn** which enables routes that are imported from the VRF-Lite for be exported out to the VPN.

After a route leaks between the VRFs, it is reoriginated and its route targets are replaced by the route target export and export map attributes specified by the new VRF's configuration.

Leaf-to-Leaf Reachability

In this example, two VPNs exist and two VRFs exist. VPN_1 is connected to VRF_A and VPN_2 is connected to VRF_B. Both VRFs are route distinguishers (RDs).

```
vrf context VRF_A
   address-family ipv4 unicast
   route-target both auto
   route-target both auto evpn
   route-target import 3:3
   route-target export 2:2
   import vrf advertise-vpn
   export vrf allow-vpn
vrf context VRF_B
   address-family ipv4 unicast
   route-target both 1:1
   route-target import 2:2
   route-target export 3:3
   import vrf advertise-vpn
   export vrf allow-vpn
```

Route leaking between the two is enabled by **allow-vpn** in an export map configured in VRF_A and VRF_B. VPN imported routes have **route-mapexport** and **export-map** attributes added to the route's set of route targets. An import map map uses the advertise-vpn option which enables routes that are imported from each VRF to be exported out to the VPN.

After a route leaks between the VRFs, it is reoriginated and its route targets are replaced by the route target export and export map attributes specified by the new VRF's configuration.

Leaf-to-Leaf with Loop Prevention

In the leaf-to-leaf configuration, you can inadvertently cause loops between the BLs that are leaking between the same VRFs unless you are careful with your route maps:

- You can use an inbound route map in each BL to deny updates from every other BL.
- If a BL originates a route, a standard community can be applied, which enables other BLs to accept the routes. This community is then stripped in the receiving BL.

In the following example, VTEPs 3.3.3.3, 4.4.4.4 and 5.5.5.5 are the BLs.

```
ip prefix-list BL_PREFIX_LIST seq 5 permit 3.3.3.3/32
ip prefix-list BL_PREFIX_LIST seq 10 permit 4.4.4.4/32
ip prefix-list BL_PREFIX_LIST seq 20 permit 5.5.5.5/32
ip community-list standard BL_COMMUNITY seq 10 permit 123:123
route-map INBOUND_MAP permit 5
  match community BL_COMMUNITY
  set community none
route-map INBOUND_MAP deny 10
  match ip next-hop prefix-list BL_PREFIX_LIST
route-map INBOUND_MAP permit 20
route-map OUTBOUND_SET_COMM permit 10
  match evpn route-type 2 mac-ip
  set community 123:123
route-map SET_COMM permit 10
  set community 123:123
route-map allow permit 10

vrf context vni100
  vni 100
  address-family ipv4 unicast
    route-target import 2:2
    route-target export 1:1
    route-target both auto
    route-target both auto evpn
    import vrf advertise-vpn
    export vrf allow-vpn
```

```
vrf context vni200
  vni 200
  address-family ipv4 unicast
    route-target import 1:1
    route-target export 2:2
    route-target both auto
    route-target both auto evpn
    import vrf advertise-vpn
    export vrf allow-vpn

router bgp 100
  template peer rr
    remote-as 100
    update-source loopback0
    address-family l2vpn evpn
      send-community
      send-community extended
      route-map INBOUND_MAP in
      route-map OUTBOUND_SET_COMM out
  neighbor 101.101.101.101
    inherit peer rr
  neighbor 102.102.102.102
    inherit peer rr
  vrf vni100
    address-family ipv4 unicast
      network 3.3.3.100/32 route-map SET_COMM
  vrf vni200
    address-family ipv4 unicast
      network 3.3.3.200/32 route-map SET_COMM
```

In this example, the tenant VRFs for the border leaf (BL) router can leak traffic by enabling extra import export flows, and the route targets in the route maps determine where the routes are imported from or exported to.

Multipath in a VRF

In this example, a VPN has multiple incoming paths. This configuration enables route leaking through an intermediate VRF, called VRF_A, which is between the VPN and another VRF, named VRF_B. Assume that multipathing is enabled in VRF_A.

```
vrf context VRF_A
   address-family ipv4 unicast
   route-target both auto evpn
   route-target export 3:3
   export vrf allow-vpn
vrf context VRF_B
   address-family ipv4 unicast
   route-target import 3:3
```

Route leaking is enabled by **allow-vpn** in the export map configured in VRF_A. When two paths for a given prefix are learnt from a VPN and imported into VRF_A, two different paths exist in VRF_B with the same source RD (VRF_A's local RD). Each route is distinguished by the original source RD (remote RD).

Path Duplication

In this example, the configuration enables a single VPN path to be imported into both VRF_A and VRF_B. Because VRF_A is configured with **export vrf allow-vpn** , VRF_A also leaks its routes into VRF_B. VRF_B then has two paths with same source RD (VRF_A's local RD), each one distinguished by the original source RD (remote RD).

```
vrf context VRF_A
   address-family ipv4 unicast
      route-target import 1:1 evpn
      route-target export 1:1 evpn
      route-target export 2:2
      export vrf allow-vpn
vrf context VRF_B
   address-family ipv4 unicast
     route-target import 1:1 evpn
     route-target import 2:2
```

This configuration creates a situation in which multipathing does not exist.

Display BGP Local Route Leaking Information

The following show commands contain information for the BGP local route leaking feature.

Command	Action
show bgp vrf *vrf-name* **process**	For a default or non-default VRF, shows the enabled state (Yes or No) of the **import advertise-vpn** and **export allow-vpn** options.
show bgp vrf *vrf-name* **ipv4 unicast** *prefix*	Shows information about imported paths, including a list of destinations a route has been imported from.

BGP Graceful Shutdown

BGP Graceful Shutdown

Beginning with release 9.3(1), BGP supports the graceful shutdown feature. This BGP feature works with the BGP **shutdown** command to:

- Dramatically decrease the network convergence time when a router or link is taken offline.
- Reduce or eliminate dropped packets that are in transit when a router or link is taken offline.

Despite the name, BGP graceful shutdown does not actually cause a shutdown. Instead, it alerts connected routers that a router or link will be going down soon.

The graceful shutdown feature uses the GRACEFUL_SHUTDOWN well-known community (0xFFFF0000 or 65535:0), which is identified by IANA and the IETF through RFC 8326. This well-known community can be attached to any routes, and it is processed like any other attribute of a route.

Because this feature announces that a router or link will be going down, the feature is useful in preparation of maintenance windows or planned outages. Use this feature before shutting down BGP to limit the impact on traffic.

Graceful Shutdown Aware and Activate

BGP routers can control the preference of all routes with the GRACEFUL_SHUTDOWN community through the concept of GRACEFUL SHUTDOWN awareness. Graceful shutdown awareness is enabled by default, which enables the receiving peers to deprefer incoming routes carrying the GRACEFUL_SHUTDOWN community. Although not a typical use case, you can disable and reenable graceful shutdown awareness through the **graceful-shutdown aware** command.

- Graceful shutdown aware is applicable only at the BGP global context. For information about contexts, see Graceful Shutdown Contexts, on page 426.
- The aware option operates with another option, the **activate** option, which you can assign to a route map for more granular control over graceful shutdown routes.

Interaction of the Graceful Shutdown Aware and Activate Options

When a graceful shutdown is activated, the GRACEFUL_SHUTDOWN community is appended to route updates only when you specify the **activate** keyword. At this point, new route updates that contain the community are generated and transmitted. When the **graceful-shutdown aware** command is configured, all routers that receive the community then deprefer (lower the route preference of) the routes in the update. Without the **graceful-shutdown aware** command, BGP does not deprefer routes with the GRACEFUL_SHUTDOWN community.

After the feature is activated and the routers are aware of graceful shutdown, BGP still considers the routes with the GRACEFUL_SHUTDOWN community as valid. However, those routes are given the lowest priority in the best-path calculation. If alternate paths are available, new best paths are chosen, and convergence occurs to accommodate the router or link that will soon go down.

Graceful Shutdown Contexts

BGP graceful shutdown feature has two contexts that determine what the feature affects and what functionality is available.

- Global context: Affects the entire switch and all routes processed by it. For example, readvertise all routes with the GRACEFUL_SHUTDOWN community.
- Peer context: Affects a BGP peer or a link between neighbors. For example, advertise only one link between peers with GRACEFUL_SHUTDOWN community.

Graceful Shutdown Contexts: Comparison of Contexts

Table 30: Comparison of Graceful Shutdown Contexts

Context	Affects	Commands
Global	The entire switch and all routes processed by it. For example, readvertise all routes with the GRACEFUL_SHUTDOWN community.	**graceful-shutdown activate [route-map** *route-map***]** **graceful-shutdown aware**
Peer	A BGP peer or a link between neighbors. For example, advertise only one link between peers with GRACEFUL_SHUTDOWN community.	**graceful-shutdown activate [route-map** *route-map***]**

Graceful Shutdown with Route Maps

Graceful shutdown works with the route policy manager (RPM) feature to control how the switch's BGP router transmits and receives routes with the GRACEFUL_SHUTDOWN community. Route maps can process route updates with the community in the inbound and outbound directions. Typically, route maps are not required. However, if needed, you can use them to customize the control of graceful shutdown routes.

Normal Inbound Route Maps

Normal inbound route maps affect routes that are incoming to the BGP router. Normal inbound route maps are not commonly used with the graceful shutdown feature because routers are aware of graceful shutdown by default.

Cisco Nexus switches running Cisco NX-OS Release 9.3(1) and later do not require an inbound route map for the graceful shutdown feature. Cisco NX-OS Release 9.3(1) and later have implicit inbound route maps that automatically deprefer any routes that have the GRACEFUL_SHUTDOWN community if the BGP router is graceful shutdown aware.

Normal inbound route maps can be configured to match against the well-known GRACEFUL_SHUTDOWN community. Although these inbound route maps are not common, there are some cases where they are used:

- If switches are running a Cisco NX-OS release earlier than 9.3(1), they do not have the implicit inbound route map present in NX-OS 9.3(1). To use the graceful shutdown feature on these switches, you must create a graceful shutdown inbound route map. The route map must match inbound routes with the well-known GRACEFUL_SHUTDOWN community, permit them, and deprefer them. If an inbound route map is needed, create it on the BGP peer that is running a version of NX-OS earlier than 9.3(1) and is receiving the graceful shutdown routes.
- If you want to disable graceful shutdown aware, but still want the router to act on incoming routes with GRACEFUL_SHUTDOWN community from some BGP neighbors, you can configure an inbound route map under the respective peers.

Normal Outbound Route Maps

Normal outbound route maps control forwarding the routes that a BGP router sends. Normal outbound route maps can affect the graceful shutdown feature. For example, you can configure an outbound route map to match on the GRACEFUL_SHUTDOWN community and set attributes, and it takes precedence over any graceful shutdown outbound route maps.

Graceful Shutdown Outbound Route Maps

Outbound Graceful shutdown route maps are specific type of outbound route map for the graceful shutdown feature. They are optional, but they are useful when you already have a community list that is associated with a route map. The typical graceful shutdown outbound route map contains only `set` clauses to set or modify certain attributes.

You can use outbound route maps in the following ways:

- For customers that already have existing outbound route maps, you can add a new entry with a higher sequence number, match on the GRACEFUL_SHUTDOWN well-known community, and add any attributes that you want.
- You can also use a graceful shutdown outbound route map with the **graceful-shutdown activate route-map** *name* option. This is the typical use case.

This route map requires no match clauses, so the route map matches on all routes being sent to the neighbor.

Route Map Precedence

When multiple route maps are present on the same router, the following order of precedence is applied to determine how routes with the community are processed:Consider the following example. Assume you have a standard outbound route map name Red that sets a local-preference of 60. Also, assume you have a peer graceful-shutdown route map that is named Blue that sets local-pref to 30. When the route update is processed, the local preference will be set to 60 because Red overwrites Blue.

- Normal outbound route maps take precedence over peer graceful shutdown maps.
- Peer graceful shutdown maps take precedence over global graceful shutdown maps.

Guidelines and Limitations

The following are limitations and guidelines for BGP global shutdown:

- Graceful shutdown feature can only help avoid traffic loss when alternative routes exist in the network for the affected routers. If the router has no alternate routes, routes carrying the GRACEFUL_SHUTDOWN community are the only ones available, and therefore, are used in the best-path calculation. This situation defeats the purpose of the feature.
- Configuring a BGP send community is required to send the GRACEFUL_SHUTDOWN community.
- For route maps:
  - When global route maps and neighbor route maps are configured, the per-neighbor route maps take precedence.
  - Outbound route maps take precedence over any global route maps configured for graceful shutdown.
  - Outbound route maps take precedence over any peer route maps configured for graceful shutdown.
  - To add the graceful shutdown functionality to legacy (existing) inbound route maps, follow this order:
    1. Add the graceful shutdown match clause to the top of the route map by setting a low sequence number for the clause (for example, sequence number 0).
    2. Add a continue statement after the graceful shutdown clause. If you omit the continue statement, route-map processing stops when it matches the graceful shutdown clause, any other clauses with higher sequence numbers (for example, 1 and higher) are not processed.

Graceful Shutdown Task Overview

To use the graceful shutdown feature, you typically enable graceful-shutdown aware on all Cisco Nexus switches and leave the feature enabled. When a BGP router must be taken offline, you configure graceful-shutdown activate on it.

The following details document the best practice for using the graceful shutdown feature.

To bring the router or link down:

1. Configure the Graceful Shutdown feature.

2. Watch the neighbor for the best path.
3. When the best path is recalculated, issue the **shutdown** command to disable BGP.
4. Perform the work that required you to shut down the router or link.

To bring the router or link back online:

1. When you finish the work that required the shutdown, reenable BGP (**no shutdown**).
2. Disable the graceful shutdown feature (**no graceful-shutdown activate** in config router mode).

Configure Graceful Shutdown on a Link

This task enables you to configure graceful shutdown on a specific link between two BGP routers.

Before you begin

If you have not already enabled BGP, enable it now (**feature bgp**).

Procedure

Step 1 **config terminal**

Example:

```
switch-1# configure terminal
                        switch-1(config)#
```

Enters global configuration mode.

Step 2 **router bgp** *autonomous-system-number*

Example:

```
switch-1(config)# router bgp 110
                        switch-1(config-router)#
```

Enters router configuration mode to create or configure a BGP routing process.

Step 3 **neighbor {** *ipv4-address*|*ipv6-address* **} remote-as** *as-number*

Example:

```
switch-1(config-router)# neighbor 10.0.0.3 remote-as 200
                        switch-1(config-router-neighbor)#
```

Configures the autonomous system (AS) to which the neighbor belongs.

Step 4 **graceful-shutdown activate [route-map** *map-name***]**

Example:

```
switch-1(config-router-neighbor)# graceful-shutdown activate route-map gshutPeer
                        switch-1(config-router-neighbor)#
```

Configures graceful shutdown on the link to the neighbor. Also, advertises the routes with the well-known GRACEFUL_SHUTDOWN community and applies the route map to the outbound route updates.

The routes are advertised with the graceful-shutdown community by default. In this example, routes are advertised to the neighbor with the Graceful-shutdown community with a route-map named gshutPeer.

The devices receiving the gshut community look at the communities of the route and optionally use the communities to apply routing policy.

Filter BGP Routes and Setting Local Preference Based On GRACEFUL_SHUTDOWN Communities

Switches that are not yet running 9.3(1) do not have an inbound route map that matches against the GRACEFUL_SHUTDOWN community name. Therefore, they have no way of identifying and depreferring the correct routes.

For switches running a release of NX-OS that is earlier than 9.3(1), you must configure an inbound route map that matches on the community value for graceful shutdown (65535:0) and deprefers routes.

If your switch is running 9.3(1) or later, you do not need to configure an inbound route map.

Procedure

Step 1 **configure terminal**

Example:

```
switch-1# configure terminal
                        switch-1<config)#
```

Enters global configuration mode.

Step 2 **ip community list standard** *community-list-name* **seq** *sequence-number* **{ permit | deny }** *value*

Example:

```
switch-1(config)# ip community-list standard GSHUT seq 10 permit 65535:0
                        switch-1(config)#
```

Configures a community list and permits or denies routes that have the well-known graceful shutdown community value.

Step 3 **route map** *map-tag* **{deny | permit}** *sequence-number*

Example:

```
switch-1(config)# route-map RM_GSHUT permit 10
                        switch-1(config-route-map)#
```

Configures a route map as sequence 10 and permits routes that have the GRACEFUL_SHUTDOWN community.

Step 4 **match community** *community-list-name*

Example:

```
switch-1(config-route-map)# match community GSHUT
                        switch-1(config-route-map)#
```

Configures that routes that match the IP community list GSHUT are processed by Route Policy Manager (RPM).

Step 5 **set local-preference** *local-pref-value*

Example:

```
switch-1(config-route-map)# set local-preference 10
                          switch-1(config-route-map)#
```

Configures that the routes that match the IP community list GSHUT will be given a specified local preference.

Step 6 **exit**

Example:

```
switch-1(config-route-map)# exit
                          switch-1(config)#
```

Leaves route map configuration and returns to global configuration mode.

Step 7 **router bgp** *community-list-name*

Example:

```
switch-1(config)# router bgp 100
                          switch-1(config-router)#
```

Enters router configuration mode and creates a BGP instance.

Step 8 **neighbor {** *ipv4-address*|*ipv6-address* **}**

Example:

```
switch-1(config-router)# neighbor 10.0.0.3
                          switch-1(config-router-neighbor)#
```

Enters route BGP neighbor mode for a specified neighbor.

Step 9 **address-family {** *address-family sub family* **}**

Example:

```
nxosv2(config-router-neighbor)# address-family ipv4 unicast
                          nxosv2(config-router-neighbor-af)#
```

Puts the neighbor into address family (AF) configuration mode.

Step 10 **send community**

Example:

```
nxosv2(config-router-neighbor-af)# send-community
                          nxosv2(config-router-neighbor-af)#
```

Enables BGP community exchange with the neighbor.

Step 11 **route map** *map-tag* **in**

Example:

```
nxosv2(config-router-neighbor-af)# route-map RM_GSHUT in
                          nxosv2(config-router-neighbor-af)#
```

Applies the route map to incoming routes from the neighbor. In this example, the route map that is named RM_GSHUT permits routes with the GRACEFUL_SHUTDOWN community from the neighbor.

Configure Graceful Shutdown for All BGP Neighbors

You can manually apply the GRACEFUL_SHUTDOWN well-known community to all the neighbors of a graceful shutdown initiator.

You can configure graceful shutdown at the global level for all BGP neighbors.

Before you begin

If you have not already enabled BGP, enable it now (**feature bgp**).

Procedure

Step 1 **configure terminal**

Example:

```
switch-1# configure terminal
                        switch-1(config)#
```

Enters global configuration mode.

Step 2 **router bgp** *autonomous-system-number*

Example:

```
switch-1(config)# router bgp 110
                        switch-1(config-router)#
```

Enters router configuration mode to create or configure a BGP routing process.

Step 3 **graceful-shutdown activate [route-map** *map-name***]**

Example:

```
switch-1(config-router-neighbor)# graceful-shutdown activate route-map gshutPeer
                        switch-1(config-router-neighbor)#
```

Configures graceful shutdown route map for the links to all neighbors. Also, advertises all routes with the well-known GRACEFUL_SHUTDOWN community and applies the route map to the outbound route updates.

The routes are advertised with the GRACEFUL_SHUTDOWN community by default. In this example, routes are advertised to all neighbors with the community with a route-map named gshutPeer. The route map should contain only set clauses.

The devices receiving the GRACEFUL_SHUTDOWN community look at the communities of the route and optionally use the communities to apply routing policy.

Control the Preference for All Routes with the GRACEFUL_SHUTDOWN Community

Before you begin

If you have not enabled BGP, enable it now (**feature bgp**).

Cisco NX-OS enables lowering the preference of incoming routes that have the GRACEFUL_SHUTDOWN community. When **graceful shutdown aware** is enabled, BGP considers routes carrying the community as the lowest preference during best path calculation. By default, lowering the preference is enabled, but you can selectively disable this option.

Whenever you enable or disable this option, you trigger a BGP best-path calculation. This option gives you the flexibility to control the behavior of the BGP best-path calculation for the graceful shutdown well-known community.

Procedure

Step 1 **configure terminal**

Example:

```
switch-1(config)# config terminal
                        switch-1(config)#
```

Enters global configuration mode.

Step 2 **router bgp** *autonoums-system*

Example:

```
switch-1(config)# router bgp 100
                        switch-1(config-router)#
```

Enters router configuration mode and configures a BGP routing process.

Step 3 (Optional) **no graceful-shutdown aware**

Example:

```
switch-1(config-router)# no graceful-shutdown aware
                        switch-1(config-router)#
```

For this BGP router, do not give lower preference for all routes that have the GRACEFUL_SHUTDOWN community. The default action is to deprefer routes when the graceful shutdown aware feature is disabled, so using the **no** form of the command is optional for not depreferring graceful shutdown routes.

Prevent Sending the GRACEFUL_SHUTDOWN Community to a Peer

If you no longer need the GRACEFUL_SHUTDOWN community that is appended as a route attribute to outbound route updates, you can remove the community, which no longer sends it to a specified neighbor. One use case would be when a router is at an autonomous system boundary, and you do not want the graceful shutdown functionality to propagate outside of an autonomous system boundary.

To prevent sending the GRACEFUL_SHUTDOWN to a peer, you can disable the send community option or strip the community from the outbound route map.

Procedure

Step 1 Disable the send-community in the running config.

Example:

```
nxosv2(config-router-neighbor-af)# no send-community standard
                        nxosv2(config-router-neighbor-af)#
```

If you use this option, the GRACEFUL_SHUTDOWN community is still received by the switch, but it is not sent to the downstream neighbor through the outbound route map. All standard communities are not sent either.

Step 2 Delete the GRACEFUL_SHUTDOWN community through an outbound route map by following these steps:

a. Create an IP community list matches the GRACEFUL_SHUTDOWN community.

b. Create an outbound route map to match against the GRACEFUL_SHUTDOWN community.

c. Use a **set community-list delete** clause to strip GRACEFUL_SHUTDOWN community.

If you use this option, the community list matches and permits the GRACEFUL_SHUTDOWN community, then the outbound route map matches against the community and then deletes it from the outbound route map. All other communities pass through the outbound route map without issue.

Display Graceful Shutdown Information

Information about the graceful shutdown feature is available through the following **show** commands.

Command	Action
show ip bgp community-list graceful-shutdown	Shows all entires in the BGP routing table that have the GRACEFUL_SHUTDOWN community.
show running-config bgp	Shows the running BGP configuration.
show running-config bgp all	Shows all information for the running BGP configuration including information about the graceful shutdown feature.
show bgp *address-family* **neighbors** *neighbor-address*	When the feature is configured for the peer, shows the following: • The state of the graceful-shutdown-activate feature for the specified neighbor • The name of any graceful shutdown route map configured for the specified neighbor

Command	Action
show bgp process	Shows different information depending on the context. When the graceful-shutdown-activate option is configured in peer context, shows the enabled or disabled state for the feature through `graceful-shutdown-active`. When the graceful-shutdown-activate option is configured in global context and has a graceful-shutdown route map, shows the enabled state of the feature through the following: • `graceful-shutdown-active` • `graceful-shutdown-aware` • `graceful-shutdown route-map`
show ip bgp *address*	For the specified address, shows the BGP routing table information, including the following: • The state of the specified address as the best path • Whether the specified address is part of the GRACEFUL_SHUTDOWN community

Graceful Shutdown Configuration Examples

These examples show some configurations for using the graceful shutdown feature.

Configuring Graceful Shutdown for a BGP Link

The following example shows how to configure graceful shutdown while setting a local preference and a community:

- Configuring graceful shutdown activate for the link to the specified neighbor
- Adding the GRACEFUL_SHUTDOWN community to the routes
- Setting a route map named gshutPeer with only set clauses for outbound routes with the community.

```
router bgp 100
    neighbor 20.0.0.3 remote-as 200
         graceful-shutdown activate route-map gshutPeer
         address-family ipv4 unicast
              send-community

route-map gshutPeer permit 10
    set local-preference 0
    set community 200:30
```

Configuring Graceful Shutdown for All-Neighbor BGP Links

The following example shows:

- Configuring graceful shutdown activate for all the links connecting the local router and all its neighbors.
- Adding the GRACEFUL_SHUTDOWN community to the routes.
- Setting a route map that is named gshutAall with only set clauses for all outbound routes.

```
router bgp 200
   graceful-shutdown activate route-map gshutAll

route-map gshutAll permit 10
   set as-path prepend 10 100 110
   set community 100:80

route-map Red permit 10
   set local-pref 20

router bgp 100
   graceful-shutdown activate route-map gshutAll
      router-id 2.2.2.2
         address-family ipv4 unicast
         network 2.2.2.2/32
         neighbor 1.1.1.1 remote-as 100
         update-source loopback0
         address-family ipv4 unicast
            send-community
         neighbor 20.0.0.3 remote-as 200
         address-family ipv4 unicast
            send-community
               route-map Red out
```

In this example, the `gshutAll` route-map takes effect for neighbor 1.1.1.1, but not neighbor 20.0.0.3, because the outbound route-map `Red` configured under neighbor 20.0.0.3 takes precedence instead.

Configuring Graceful Shutdown Under a Peer-Template

This example configures the graceful shutdown feature under a peer-session template, which is inherited by a neighbor.

```
router bgp 200
   template peer-session p1
      graceful-shutdown activate route-map gshut_out
   neighbor 1.1.1.1 remote-as 100
      inherit peer-session p1
      address-family ipv4 unicast
         send-community
```

Filtering BGP Routes and Setting Local Preference Based on GRACEFUL_SHUTDOWN Community Using and Inbound Route Map

This example shows how to use a community list to filter the incoming routes that have the GRACEFUL_SHUTDOWN community. This configuration is useful for legacy switches that are not running Cisco NX-OS 9.3(1) as a minimum version.

The following example shows:

- An IP Community List that permits routes that have the GRACEFUL_SHUTDOWN community.
- A route map that is named RM_GSHUT that permits routes based on a standard community list named GSHUT.

- The route map also sets the preference for the routes it processes to 0 so that those routes are given lower preference for best path calculation when the router goes offline. The route map is applied to incoming IPv4 routes from the neighbor (20.0.0.2).

```
ip community-list standard GSHUT permit 65535:0

route-map RM_GSHUT permit 10
   match community GSHUT
   set local-preference 0

router bgp 200
   neighbor 20.0.0.2 remote-as 100
      address-family ipv4 unicast
         send-community
         route-map RM_GSHUT in
```

Configure a Graceful Restart

You can configure a graceful restart and enable the graceful restart helper feature for BGP.



Note Cisco NX-OS Release 10.1(1) supports a higher number of BFD sessions. If BGP sessions are associated with BFD, the BGP **restart-time** may need to be increased to maintain peer connection during ISSU.



Note From the perspective of BGP Graceful Restart, if there are idle peers during a node restart, they can potentially cause traffic loss during an ISSU because they may delay the establishment of the first best-path. It is recommended to either bring all these idle neighbors up, or configure 'shutdown' under each of them, or remove them entirely from the configuration.

Before you begin

You must enable BGP (see the "Enabling BGP" section).

Create the VRFs.

Procedure

Step 1 **configure terminal**

Example:

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

Enters configuration mode.

Step 2 **router bgp** *as-number*

Example:

```
switch(config)# router bgp 65535
switch(config-router)#
```

Creates a new BGP process with the configured autonomous system number.

Step 3 (Optional) **timers prefix-peer-timeout** *timeout*

Example:

```
switch(config-router)# timers prefix-peer-timeout 20
```

Configures the timeout value (in seconds) for BGP prefix peers. The default value is 90 seconds.

Note This command is supported beginning with Cisco NX-OS Release 9.3(3).

Step 4 **graceful-restart**

Example:

```
switch(config-router)# graceful-restart
```

Enables a graceful restart and the graceful restart helper functionality. This command is enabled by default.

This command triggers an automatic notification and session reset for the BGP neighbor sessions.

Step 5 **graceful-restart** {**restart-time** *time*|**stalepath-time** *time*}

Example:

```
switch(config-router)# graceful-restart
restart-time 300
```

Configures the graceful restart timers.

The optional parameters are as follows:

- **restart-time** —Maximum time for a restart sent to the BGP peer. The range is from 1 to 3600 seconds. The default is 120.

  Note Cisco NX-OS Release 10.1(1) supports a higher number of BFD sessions. If BGP sessions are associated with BFD, the BGP **restart-time** may need to be increased to maintain peer connection during ISSU.

- **stalepath-time** —Maximum time that BGP keeps the stale routes from the restarting BGP peer. The range is from 1 to 3600 seconds. The default is 300.

In NX-OS software release 10.2(1), a manual reset of a BGP session is needed for the BGP session to advertise Graceful Restart capabilities. For NX-OS software releases 10.2(2) and later, BGP sessions dynamically advertise Graceful Restart capabilities without needing to restart the BGP sessions when this command is enabled.

Step 6 **graceful-restart-helper**

Example:

```
switch(config-router)# graceful-restart
restart-time 300
```

With BGP GR disabled, the N9K itself will not necessarily preserve its own forwarding state during certain GR-capable events like SSO, BGP process restart, etc. occurring locally on the N9K. However, as a GR helper, it will support a peer that has advertised its GR capability and is restarting. This means, when the N9K detects the peering has gone down

(other than a holdtimer expiration or receipt of a Notification message), the N9K will stale the routes pointing to the peer and will wait for the peer's EOR (or stalepath timeout). When the peer restarts and re-establishes its peering with the N9K, it will re-advertise all its own routes and the N9K will refresh them in its BGP and routing tables. On receipt of the EOR from the peer or the stalepath timeout (whichever occurs first), the N9K will flush any remaining stale routes from that peer. In the absence of helper mode, the N9K would instantly clear out the routes learnt from the remote peer that was restarting which could lead to traffic loss.

Step 7 (Optional) **show running-config bgp**

Example:

```
switch(config-router)# show
running-config bgp
```

Displays the BGP configuration.

Step 8 (Optional) **copy running-config startup-config**

Example:

```
switch(config-router)# copy
running-config startup-config
```

Saves this configuration change.

This example shows how to enable a graceful restart:

```
switch# configure terminal
 switch(config)# router bgp 65536
 switch(config-router)# graceful-restart
 switch(config-router)# graceful-restart restart-time 300
 switch(config-router)# copy running-config startup-config
```

Configure Virtualization

You can create multiple VRFs within each VDC and use the same BGP process in each VRF.

You can configure one BGP process in each VDC. You can create multiple VRFs within each VDC and use the same BGP process in each VRF.

You can configure one BGP process, create multiple VRFs, and use the same BGP process in each VRF.

Before you begin

Ensure that you have enabled the BGP feature

- You must enable BGP (see the Enabling BGP section).
- Ensure that you are in the correct VDC (or use the **switchto vdc** command).

You must enable BGP.

Procedure

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 **exit**

Example:

```
switch(config-vrf)# exit
  switch(config)#
```

Exits VRF configuration mode.

Step 4 **router bgp** *as-number*

Example:

```
switch(config)# router bgp 65535
  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 65535
  switch(config-router--vrf-neighbor)#
```

Configures the IP address and AS number for a remote BGP peer.

Step 7 (Optional) **copy running-config startup-config**

Example:

```
switch(config-router-vrf-neighbor)# copy
  running-config startup-config
```

Saves this configuration change.

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
```

Verify the Advanced BGP Configuration

To display the BGP configuration, 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 {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **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*] {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **community-list** *list-name* [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP community list.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **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 {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **extcommunity-list** *list-name* [**exact-match**]} [**vrf** vrf-name]	Displays the BGP routes that match a BGP extended community list.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **extcommunity-list** *list-name* [**exact-match**]} [**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 {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] {**dampening dampened-paths** [**regexp** *expression*]} [**vrf** vrf-name]	Displays the BGP route history paths.
show bgp {**ipv4** | **ipv6** | **vpnv4** | **vpnv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **filter-list** *list-name* [**vrf** *vrf-name*]	Displays the information for the BGP filter list.

Command	Purpose
show bgp {**ipv4** | **ipv6** | **vpnv4** | **vpnv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **neighbors** [*ip-address* | *ipv6-prefix*] [**vrf** *vrf-name*]	Displays the information for BGP peers. Use the **clear bgp neighbors** command to clear these neighbors.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] {**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 {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **policy** *name* [**vrf** *vrf-name*]	Displays the BGP policy information. Use the **clear bgp policy** command to clear the policy information.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **prefix-list** *list-name* [**vrf** *vrf-name*]	Displays the BGP routes that match the prefix list.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **received-paths** [**vrf** *vrf-name*]	Displays the BGP paths stored for soft reconfiguration.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **regexp** *expression* [**vrf** *vrf-name*]	Displays the BGP routes that match the AS_path regular expression.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **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 bgp {**ipv4** | **ipv6**} **unicast neighbors** *interface*	Displays information about BGP peers for the specified interface.
show ip bgp neighbors *interface-name*	Displays the interface used as a BGP peer.

Command	Purpose
show ip route *ip-address* **detail vrf all | i bw**	Displays the link bandwidth EXTCOMM fields. bw:xx (such as bw:40) in the output indicates that BGP peers are sending BGP extended attributes with the bandwidth (for weighted ECMP).
show {**ipv4** | **ipv6**} **bgp** *options*	Displays the BGP status and configuration information.
show {**ipv4** | **ipv6**} **mbgp** *options*	Displays the BGP status and configuration information.
show ipv6 routers interface *interface*	Displays the link-local address of remote IPv6 routers, which is learned through IPv6 ICMP router advertisement.
show running-configuration bgp	Displays the current running BGP configuration.

Monitor BGP Statistics

To display BGP statistics, use the following commands:

Command	Purpose
show bgp {**ipv4** | **vpnv4** | **vpnv6** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **flap-statistics** [**vrf** *vrf-name*] **show bgp** {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **flap-statistics** [**vrf** *vrf-name*]	Displays the BGP route flap statistics. Use the **clear bgp flap-statistics** command to clear these statistics.
show bgp {**ipv4** | **ipv6** | **vpnv4** | **vpnv6**} **unicast injected-routes** **show bgp** {**ipv4** | **ipv6**} **unicast injected-routes**	Displays injected routes in the routing table.
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

This example shows how to enable BFD for individual BGP neighbors:

```
router bgp 400
    router-id 2.2.2.2
```

```
    neighbor 172.16.2.3
    bfd
    remote-as 400
    update-source Vlan1002
    address-family ipv4 unicast
```

This example shows how to enable BFD for BGP prefix peers:

```
router bgp 400
    router-id 1.1.1.1
    neighbor 172.16.2.0/24
    bfd
    remote-as 400
    update-source Vlan1002
    address-family ipv4 unicast
```

This example shows how to configure MD5 authentication for prefix-based neighbors:

```
template peer BasePeer-V6
    description BasePeer-V6
    password 3 f4200cfc725bbd28
    transport connection-mode passive
    address-family ipv6 unicast
    template peer BasePeer-V4
    bfd
    description BasePeer-V4
    password 3 f4200cfc725bbd28
    address-family ipv4 unicast
    --
    neighbor fc00::10:3:11:0/127 remote-as 65006
    inherit peer BasePeer-V6
    neighbor 10.3.11.0/31 remote-as 65006
    inherit peer BasePeer-V4
```

This example shows how to enable neighbor status change messages globally and suppress them for a specific neighbor:

```
router bgp 65100
    log-neighbor-changes
    neighbor 209.165.201.1 remote-as 65535
    description test
    address-family ipv4 unicast
    soft-reconfiguration inbound
    disable log-neighbor-changes
```

Related Topics

The following topics can give more information on BGP:

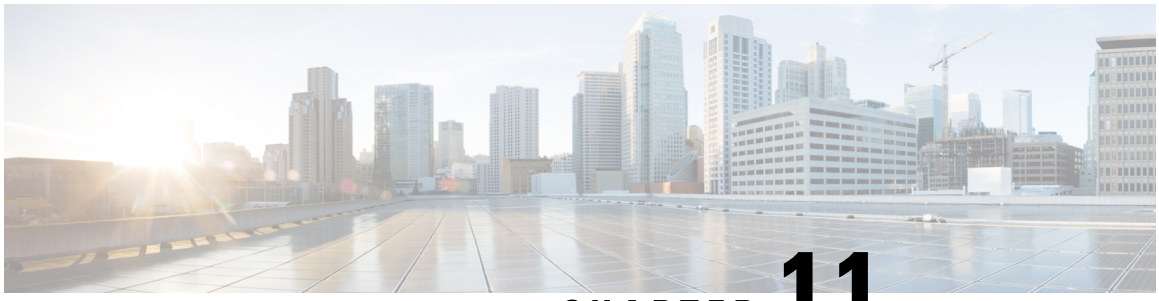
- *Configuring Basic BGP*
- *Configuring Route Policy Manager*

Additional References

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

MIBs

MIBs	MIBs Link
MIBs related to BGP	To locate and download supported MIBs, go to the following URL: https://cisco.github.io/cisco-mibs/supportlists/nexus9000/Nexus9000MIBSupportList.html



CHAPTER 11

Configure RIP

- RIP, on page 447
- Prerequisites for RIP, on page 449
- Guidelines and Limitations for RIP, on page 449
- Default Settings for RIP Parameters, on page 450
- RIPs, on page 450
- Verify the RIP Configuration, on page 462
- Display the RIP Statistics, on page 463
- Configuration Examples for RIP, on page 463
- Related Topics, on page 464

RIP

RIP

RIP uses User Datagram Protocol (UDP) data packets to exchange routing information in small internetworks. RIPv2 supports IPv4 and IPv6. RIPv2 uses an optional authentication feature supported by the RIPv2 protocol (see the RIPv2 Authentication section).

RIP Message Types

- Request—Sent to the multicast address 224.0.0.9 to request route updates from other RIP-enabled routers.
- Response—Sent every 30 seconds by default (see the Verifying the RIP Configuration section). The router also sends response messages after it receives a request message. The response message contains the entire RIP route table. RIP sends multiple response packets for a request if the RIP routing table cannot fit in one response packet.

RIPv2 Authentication

You can configure authentication on RIP messages to prevent unauthorized or invalid routing updates in your network. Cisco NX-OS supports a simple password or an MD5 authentication digest.

You can configure the RIP authentication per interface by using keychain management for the authentication keys. Keychain management allows you to control changes to the authentication keys used by an MD5

authentication digest or simple text password authentication. See the Cisco Nexus 9000 Series NX-OS Security Configuration Guide for more details about creating keychains.

To use an MD5 authentication digest, you configure a password that is shared at the local router and all remote RIP neighbors. Cisco NX-OS creates an MD5 one-way message digest based on the message itself and the encrypted password and sends this digest with the RIP message (Request or Response). The receiving RIP neighbor validates the digest by using the same encrypted password. If the message has not changed, the calculation is identical, and the RIP message is considered valid.

An MD5 authentication digest also includes a sequence number with each RIP message to ensure that no message is replayed in the network.

Split Horizons

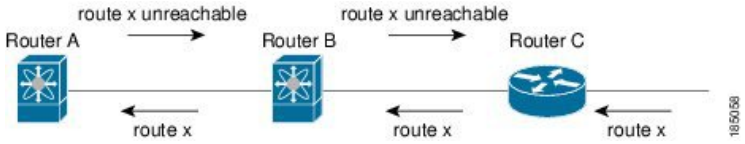
You can use split horizon to ensure that RIP never advertises a route out of the interface where it was learned.

Split horizon is a method that controls the sending of RIP update and query packets. When you enable split horizon on an interface, Cisco NX-OS does not send update packets for destinations that were learned from this interface. Controlling update packets in this manner reduces the possibility of routing loops.

You can use split horizon with poison reverse to configure an interface to advertise routes learned by RIP as unreachable over the interface that learned the routes.

The following figure shows a sample RIP network with split horizon and poison reverse enabled.

Figure 36: RIP with Split Horizon Poison Reverse



Router C learns about route X and advertises that route to Router B. Router B in turn advertises route X to Router A but sends a route X unreachable update back to Router C.

By default, split horizon is enabled on all interfaces.

Route Filtering

You can configure a route policy on a RIP-enabled interface to filter the RIP updates. Cisco NX-OS updates the route table with only those routes that the route policy allows.

Route Summarizations

You can configure multiple summary aggregate addresses 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, RIP advertises the summary address from the interface with a metric equal to the maximum metric of the more specific routes.

Note Cisco NX-OS does not support automatic route summarization.

Route Redistributions

You can use RIP to redistribute static routes or routes from other protocols. You must configure a route map with the redistribution to control which routes are passed into RIP. A route policy allows you to filter routes based on attributes such as the destination, origination protocol, route type, route tag, and so on. For more information, see Configuring Route Policy Manager, on page 523.

Whenever you redistribute routes into a RIP routing domain, Cisco NX-OS does not, by default, redistribute the default route into the RIP routing domain. You can generate a default route into RIP, which can be controlled by a route policy.

You also configure the default metric that is used for all imported routes into RIP.

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 usage of network segments and increases effective network bandwidth.

Cisco NX-OS supports the Equal Cost Multiple Paths (ECMP) feature with up to 16 equal-cost paths in the RIP route table and the unicast RIB. You can configure RIP to load balance traffic across some or all of those paths.

High Availability for RIP

Cisco NX-OS supports stateless restarts for RIP. After a reboot or supervisor switchover, Cisco NX-OS applies the running configuration, and RIP immediately sends request packets to repopulate its routing table.

Virtualization Support for RIP

Cisco NX-OS supports multiple instances of the RIP protocol that run on the same system. RIP supports virtual routing and forwarding (VRF) instances.

Prerequisites for RIP

RIP has the following prerequisites:

- You must enable RIP (see the Enabling RIP section).

Guidelines and Limitations for RIP

RIP Configuration Guidelines and Limitations

Follow these guidelines and limitations when configuring RIP.

- 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 not two different entries.
- Cisco NX-OS does not support RIPv1. If Cisco NX-OS receives an RIPv1 packet, it logs a message and drops the packet.
- Cisco NX-OS does not establish adjacencies with RIPv1 routers.
- RIP is not supported in tunnel interfaces.
- RIP only supports an 8-bit KeyID, that is less than or equal to 255. This is the keyID used while configuring authentication with RIP.
- Beginning with Cisco NX-OS Release 10.6(2)F, RIP is supported on Cisco Nexus N9324C-SE1U and N9348Y2C6D-SE1U platform switches.

Default Settings for RIP Parameters

This section lists the default settings for RIP parameters.

Table 31: Default RIP Parameters

Parameters	Default
Maximum paths for load balancing	16
RIP feature	Disabled
Split horizon	Enabled

RIPs

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.

Enable RIP

You must enable RIP before you can configure RIP.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Enters global configuration mode.

Step 2 (Optional) Use the [**no**] **feature rip** command to enable the RIP feature.

Example:

```
switch(config)# feature rip
```

Enables the RIP feature.

Step 3 (Optional) Use the **show feature** command to display enabled and disabled features.

Example:

```
switch(config)# show feature
```

Displays enabled and disabled features.

Step 4 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config)# copy running-config
startup-config
```

Saves this configuration change.

Create a RIP Instance

Before you begin

You must enable RIP (see the Enabling RIP section).

You can create a RIP instance and configure the address family for that instance.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 (Optional) Use the **[no] router rip** *instance-tag* command to create a new RIP instance with the configured *instance-tag*.

Example:

```
switch(config)# router RIP Enterprise
switch(config-router)#
```

Step 3 (Optional) Use the **address-family** {**ipv4** | **ipv6**} **unicast** command to configure the address family for this RIP instance and enter address-family configuration mode.

Example:

```
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)#
```

Step 4 (Optional) Use the **show ip rip** [**instance** *instance-tag*] [**vrf** *vrf-name*] command to display a summary of RIP information for all RIP instances.

Example:

```
switch(config-router-af)# show ip rip
```

Step 5 (Optional) Use the **distance** *value* command to set the administrative distance for RIP. The range is from 1 to 255. The default is 120. See the Administrative Distance section.

Example:

```
switch(config-router-af)# distance 30
```

Step 6 (Optional) Use the **maximum-paths** *number* command to configure the maximum number of equal-cost paths that RIP maintains in the route table. The range is from 1 to 64. The default is 16.

Example:

```
switch(config-router-af)# maximum-paths 6
```

Step 7 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-router-af)# copy running-config
startup-config
```

This example shows how to create a RIP instance for IPv4 and set the number of equal-cost paths for load balancing:

```
switch# configure terminal
switch(config)# router rip Enterprise
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# max-paths 10
switch(config-router-af)# copy running-config startup-config
```

Restart the RIP Instance

You can restart a RIP instance and remove all associated neighbors for the instance.

To restart a RIP instance and remove all associated neighbors, use the following command in global configuration mode:

Procedure

(Optional) Use the **restart rip** *instance-tag* command.

Example:

```
switch(config)# restart rip Enterprise
```

Restarts the RIP instance and removes all neighbors.

Configure RIP on an Interface

Before you begin

You must enable RIP (see the Enabling RIP section).

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 (Optional) Use the **interface** *interface-type slot*/*port* command to enter interface configuration mode.

Example:

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

Step 3 (Optional) Use the **ip router rip** *instance-tag* command to associate this interface with a RIP instance.

Example:

```
switch(config-if)# ip router rip
Enterprise
```

Step 4 (Optional) Use the **show ip rip** [**instance** *instance-tag*] **interface** [*interface-type slot*/*port*] [**vrf** *vrf-name*] [**detail**] command to display RIP information for an interface.

Example:

```
switch(config-if)# show ip rip
Enterprise tethernet 1/2
```

Step 5 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config
startup-config
```

This example shows how to add Ethernet 1/2 interface to a RIP instance:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ip router rip Enterprise
switch(config)# copy running-config startup-config
```

Configure RIP Authentication

Before you begin

You must enable RIP (see the Enabling RIP section).

Configure a keychain if necessary before enabling authentication. For details about implementing keychains, see the Cisco Nexus 9000 Series NX-OS Security Configuration Guide.

You can configure authentication for RIP packets on an interface.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 (Optional) Use the **interface** *interface-type slot*/*port* command to enter interface configuration mode.

Example:

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

Step 3 (Optional) Use the **ip rip authentication mode** {**text** | **md5**} command to set the authentication type for RIP on this interface as cleartext or MD5 authentication digest.

Example:

```
switch(config-if)# ip rip authentication
mode md5
```

Step 4 (Optional) Use the **ip rip authentication key-chain** *key* command to configure the authentication key used for RIP on this interface.

Example:

```
switch(config-if)# ip rip authentication key-chain RIPKey
```

Step 5 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config
startup-config
```

This example shows how to create a keychain and configure MD5 authentication on a RIP interface:

```
switch# configure terminal
switch(config)# key chain RIPKey
switch(config-keychain)# key 2
switch(config-keychain-key)# accept-lifetime 00:00:00 Jan 01 2000 infinite
switch(config-keychain-key)# send-lifetime 00:00:00 Jan 01 2000 infinite
switch(config-keychain-key)# exit
switch(config-keychain)# exit
switch(config)# interface ethernet 1/2
switch(config-if)# ip rip authentication mode md5
switch(config-if)# ip rip authentication key-chain RIPKey
switch(config-if)# copy running-config startup-config
```

Configure a Passive Interface

You can configure a RIP interface to receive routes but not send route updates by setting the interface to passive mode.

To configure a RIP interface in passive mode, use the following command in interface configuration mode:

Procedure

(Optional) Use the **ip rip passive-interface** command.

Example:

```
switch(config-if)# ip rip
passive-interface
```

Sets the interface to passive mode.

Configure Split Horizon With Poison Reverse

You can configure an interface to advertise routes learned by RIP as unreachable over the interface that learned the routes by enabling poison reverse.

To configure split horizon with poison reverse on an interface, use the following command in interface configuration mode:

Procedure

(Optional) Use the **ip rip poison-reverse** command.

Example:

```
switch(config-if)# ip rip poison-reverse
```

Enables split horizon with poison reverse. Split horizon with poison reverse is disabled by default.

Configure Route Summarization

You can create aggregate addresses that are represented in the routing table by a summary address. Cisco NX-OS advertises the summary address metric that is the smallest metric of all the more specific routes.

To configure a summary address on an interface, use the following command in interface configuration mode:

Procedure

(Optional) Use the {**ip** | **ipv6**}**rip summary-address***ip-prefix/mask-len* command.

Example:

```
switch(config-if)# ip rip summary-address 1.1.1.1/32
```

Configures a summary address for RIP for IPv4 or IPv6 addresses.

Configure Route Redistribution

Before you begin

You must enable RIP (see the Enabling RIP section)

Configure a route map before configuring redistribution . See the Configuring Route Maps section for details on configuring route maps.

You can configure RIP to accept routing information from another routing protocol and redistribute that information through the RIP network. Redistributed routes can be assigned a default route.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 (Optional) Use the **router rip** *instance-tag* command to create a new RIP instance with the configured *instance-tag*.

Example:

```
switch(config)# router rip Enterprise
switch(config-router)#
```

Step 3 (Optional) Use the **address-family** {**ipv4** | **ipv6**} **unicast** command to enter address-family configuration mode.

Example:

```
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)#
```

Step 4 (Optional) Use the **redistribute** {**bgp** *as* | **direct** | {**eigrp** | **isis** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static**} **route-map** *map-name* command to redistribute routes from other protocols into RIP.

Example:

```
switch(config-router-af)# redistribute
eigrp 201 route-map RIPmap
```

Step 5 (Optional) Use the **default-information originate** [**always**] [**route-map** *map-name*] command to generate a default route into RIP, controlled by a route map if required.

Example:

```
switch(config-router-af)#
default-information originate always
```

Step 6 (Optional) Use the **default-metric** *value* command to set the default metric for all redistributed routes.

Example:

```
switch(config-router-af)# default-metric
2
```

Step 7 (Optional) Use the **show ip rip route** [*ip-prefix* [**longer-prefixes** | **shorter-prefixes**]] [**vrf** *vrf-name*] [**summary**] command to show the routes in RIP.

Example:

```
switch(config-router-af)# show ip rip
route
```

Step 8 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-router-af)# copy running-config
startup-config
```

This example shows how to redistribute EIGRP into RIP:

```
switch# configure terminal
switch(config)# router rip Enterprise
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# redistribute eigrp 201 route-map RIPmap
switch(config-router-af)# copy running-config startup-config
```

Configure Cisco NX-OS RIP for Compatibility with Cisco IOS RIP

Before you begin

You must enable RIP (see the Enabling RIP section).

You can configure Cisco NX-OS RIP to behave like Cisco IOS RIP in the way that routes are advertised and processed.

Directly connected routes are treated with cost 1 in Cisco NX-OS RIP and with cost 0 in Cisco IOS RIP. When routes are advertised in Cisco NX-OS RIP, the receiving device adds a minimum cost of +1 to all received routes and installs the routes in its routing table. In Cisco IOS RIP, this cost increment is done on the sending router, and the receiving router installs the routes without any modification. This difference in behavior can cause issues when both Cisco NX-OS and Cisco IOS devices are working together. You can prevent these compatibility issues by configuring Cisco NX-OS RIP to advertise and process routes like Cisco IOS RIP.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Enters global configuration mode.

Step 2 Configure the RIP instance and compatibility settings.

a) Use the **router rip** *instance-tag* command to create a new RIP instance with the configured instance tag.

Example:

```
switch(config)# router rip 100
switch(config-router)#
```

Creates a new RIP instance with the configured instance tag. You can enter 100, 201, or up to 20 alphanumeric chapters for the instance tag.

b) Use the [**no**] **metric direct 0** command to configure all directly connected routes with cost 0 for Cisco IOS RIP compatibility.

Example:

```
switch(config-router)# metric direct 0
```

Configures all directly connected routes with cost 0 instead of the default of cost 1 in order to make Cisco NX-OS RIP compatible with Cisco IOS RIP in the way that routes are advertised and processed.



Note This command must be configured on all Cisco NX-OS devices that are present in any RIP network that also contains Cisco IOS devices.

c) Use the **address-family ipv4 unicast** command to configure the VRF address family for this RIP instance.

Example:

```
switch(config-router)# address-family ipv4 unicast
```

Configures the VRF address family for this RIP instance.

d) Use the **vrf abc** command to create a new VRF.

Example:

```
switch(config-router)# vrf abc
```

Creates a new VRF.

e) Use the **flush-routes** command to remove routes.

Example:

```
switch(config-router)# flush-routes
```

Remove routes.

f) Use the **isolate** command to isolate the router from RIP perspective.

Example:

```
switch(config-router)# isolate
```

Isolates router from RIP perspective.

g) Use the **shutdown** command to shut down the RIP instance.

Example:

```
switch(config-router)# shutdown
```

Shuts down the RIP instance.

Note The show run output will differ if the shutdown command configured is under process and a vrf is also created. The shutdown configuration will be shown above vrf configuration.

Step 3 (Optional) Use the **show running-config rip** command to display the current running RIP configuration.

Example:

```
switch(config-router)# show
running-config rip
!Command: show running-config rip
!Running configuration last done at: Tue Dec 17 01:13:28 2024
!Time: Tue Dec 17 01:13:31 2024
```

Displays the current running RIP configuration.

Step 4 (Optional) Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-router)# copy running-config
startup-config
```

Saves this configuration change.

This example shows how to disable Cisco NX-OS RIP compatibility with Cisco IOS RIP by returning all direct routes from cost 0 to cost 1:

```
switch# configure terminal
switch(config)# router rip 100
switch(config-router)# no metric direct 0
switch(config-router)# show running-config rip
switch(config-router)# copy running-config startup-config
```

Configure Virtualization

Before you begin

You must enable RIP (see the Enabling RIP section).kaanipakaa

You can configure multiple RIP instances, create multiple VRFs, and use the same or multiple RIP instances in each VRF. You assign a RIP 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 configurations for that interface.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 Configure the VRF context and RIP instance.

a) (Optional) Use the **vrf context** *vrf-name* command to create a new VRF and enter VRF configuration mode.

Example:

```
switch(config)# vrf context RemoteOfficeVRF
switch(config-vrf)#
```

b) (Optional) Use the **exit** command to exit VRF configuration mode.

Example:

```
switch(config-vrf)# exit
switch(config)#
```

c) (Optional) Use the **router rip** *instance-tag* command to create a new RIP instance with the configured instance tag.

Example:

```
switch(config)# router rip Enterprise
switch(config-router)#
```

d) (Optional) Use the **vrf** *vrf-name* command to create a new VRF.

Example:

```
switch(config-router)# vrf RemoteOfficeVRF
switch(config-router-vrf)#
```

e) (Optional) Use the **address-family** {**ipv4** | **ipv6**} **unicast** command to configure the VRF address family for this RIP instance.

Example:

```
switch(config-router-vrf)# address-family ipv4 unicast
switch(config-router-vrf-af)#
```

f) (Optional) Use the **redistribute** {**bgp** *as* | **direct** | {**eigrp** | **isis** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static**} **route-map** *map-name* command to redistribute routes from other protocols into RIP.

Example:

```
switch(config-router-vrf-af)# redistribute eigrp 201 route-map RIPmap
```

See Configure route maps, on page 550 for more information about route maps.

Step 3 Configure the interface and assign it to the VRF.

a) (Optional) Use the **interface ethernet** *slot*/*port* command to enter interface configuration mode.

Example:

```
switch(config-router-vrf-af)# interface ethernet 1/2
switch(config-if)#
```

b) (Optional) Use the **vrf member** *vrf-name* command to add this interface to a VRF.

Example:

```
switch(config-if)# vrf member RemoteOfficeVRF
```

c) (Optional) Use the **ip address** *ip-prefix*/*length* command to configure an IP address for this interface.

Example:

```
switch(config-if)# ip address 192.0.2.1/16
```

You must perform this step after you assign this interface to a VRF.

d) (Optional) Use the **ip router rip** *instance-tag* command to associate this interface with a RIP instance.

Example:

```
switch(config-if)# ip router rip Enterprise
```

Step 4 (Optional) Use the **show ip rip** [**instance** *instance-tag*] **interface** [*interface-type slot*/*port*] [**vrf** *vrf-name*] command to display RIP information for an interface in a VRF.

Example:

```
switch(config-if)# show ip rip Enterprise ethernet 1/2
```

Step 5 Use the **copy running-config startup-config** command to save this configuration change.

Example:

```
switch(config-if)# copy running-config startup-config
```

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 rip Enterprise
switch(config-router)# vrf RemoteOfficeVRF
switch(config-router-vrf)# address-family ipv4 unicast
switch(config-router-vrf-af)# redistribute eigrp 201 route-map RIPmap
switch(config-router-vrf-af)# 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 rip Enterprise
switch(config-if)# copy running-config startup-config
```

Tune the RIP Protocol

You can tune RIP to match your network requirements. RIP uses several timers that determine the frequency of routing updates, the length of time before a route becomes invalid, and other parameters. You can adjust these timers to tune routing protocol performance to better suit your internetwork needs.

Note You must configure the same values for the RIP timers on all RIP-enabled routers in your network.

Procedure

Step 1 (Optional) Use the **timers basic** *update timeout holddown garbage-collection* command to set the RIP timers in seconds.

Example:

```
switch(config-router-af)# timers basic 40 120 120 100
```

The parameters are as follows:

- *update*—The range is from 5 to any positive integer. The default is 30.
- *timeout*—The time that Cisco NX-OS waits before declaring a route as invalid. If Cisco NX-OS does not receive route update information for this route before the timeout interval ends, Cisco NX-OS declares the route as invalid. The range is from 1 to any positive integer. The default is 180.
- *holddown*—The time during which Cisco NX-OS ignores better route information for an invalid route. The range is from 0 to any positive integer. The default is 180.
- *garbage-collection*—The time from when Cisco NX-OS marks a route as invalid until Cisco NX-OS removes the route from the routing table. The range is from 1 to any positive integer. The default is 120.

Step 2 (Optional) Use the **ip rip metric-offset** *value* command to add a value to the metric for every route received on this interface.

Example:

```
switch(config-if)# ip rip metric-offset 10
```

The range is from 1 to 15. The default is 1.

Step 3 (Optional) Use the **ip rip route-filter** {**prefix-list** *list-name* | **route-map** *map-name* | [**in** | **out**]} command to specify a route map to filter incoming or outgoing RIP updates.

Example:

```
switch(config-if)# ip rip route-filter route-map
            InputMap in
```

Verify the RIP Configuration

Procedure

Step 1 Use the **show ip rip instance** [*instance-tag*] [**vrf** *vrf-name*] command to display the status for an instance of RIP.

Example:

```
show ip rip instance [instance-tag] [vrf vrf-name]
```

Step 2 Use the **show ip rip** [**instance** *instance-tag*] **interface** *slot/port* **detail** [**vrf** *vrf-name*] command to display the RIP status for an interface.

Example:

```
show ip rip [instance instance-tag] interface slot/port detail [vrf vrf-name]
```

Step 3 Use the **show ip rip** [**instance** *instance-tag*] **neighbor** [*interface-type number*] [**vrf** *vrf-name*] command to display the RIP neighbor table.

Example:

```
show ip rip [instance instance-tag] neighbor [interface-type number] [vrf vrf-name]
```

Step 4 Use the **show ip rip** [**instance** *instance-tag*] **route** [*ip-prefix/length* [**longer-prefixes** | **shorter-prefixes**]] [**summary**] [**vrf** *vrf-name*] command to display the RIP route table.

Example:

```
show ip rip [instance instance-tag] route [ip-prefix/length [longer-prefixes | shorter-prefixes]]
[summary] [vrf vrf-name]
```

Step 5 Use the **show running-configuration rip** command to display the current running RIP configuration.

Example:

```
show running-configuration rip
```

Display the RIP Statistics

Procedure

Step 1 Use the **show ip rip** [**instance** *instance-tag*] **policy statistics redistribute** {**bgp** *as* | **direct** | {**eigrp** | **isis** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static**} [**vrf** *vrf-name*] command to display the RIP policy statistics.

Example:

```
switch# show ip rip [instance instance-tag] policy statistics redistribute {bgp as | direct | {eigrp
 | isis | ospf | ospfv3 | rip} instance-tag | static} [vrf vrf-name]
RIP Policy Statistics Output...
```

Step 2 Use the **show ip rip** [**instance** *instance-tag*] **statistics** *interface-type number* [**vrf** *vrf-name*] command to display the RIP statistics.

Example:

```
switch# show ip rip [instance instance-tag] statistics interface-type number [vrf vrf-name]
RIP Statistics Output...
```

Step 3 Use the **clear rip policy statistics redistribute** *protocol process-tag* command to clear policy statistics.

Example:

```
switch# clear rip policy statistics redistribute protocol process-tag
Policy statistics cleared.
```

Step 4 Use the **clear ip rip statistics** command to clear RIP statistics.

Example:

```
switch# clear ip rip statistics
RIP statistics cleared.
```

Configuration Examples for RIP

Configuration Examples for RIP.

Configuration Examples

The following example shows how to create the Enterprise RIP instance in a VRF and add Ethernet interface 1/2 to this RIP instance. The example also shows how to configure authentication for Ethernet interface 1/2 and redistribute EIGRP into this RIP domain.

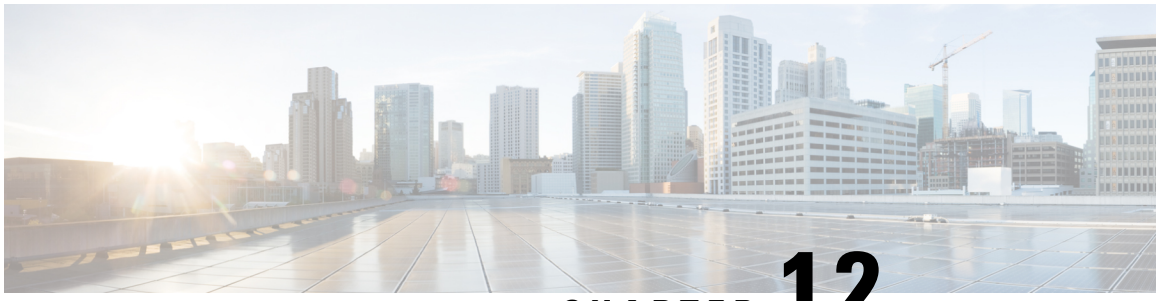
```
vrf context NewVRF
!
 feature rip
 router rip Enterprise
  vrf NewVRF
   address-family ipv4 unicast
    redistribute eigrp 201 route-map RIPmap
    maximum-paths 10
!
interface ethernet 1/2
 vrf member NewVRF
 ip address 192.0.2.1/16
 ip router rip Enterprise
 ip rip authentication mode md5
 ip rip authentication key-chain RIPKey
```

The following example shows a valid keyID configuration:

```
### Valid
key-chain kc1
key 255
key-string ...
```

Related Topics

See Configuring Route Policy Manager, on page 523 for more information on route maps.



CHAPTER 12

Configure RIPng

- RIPng, on page 465
- Prerequisites for RIPng, on page 467
- Guidelines and limitations for RIPng, on page 467
- Default settings for RIPng parameters, on page 468
- Configure RIPng, on page 468
- Verify the RIPng configuration, on page 477
- Display RIPng statistics, on page 478
- Configuration Examples for RIPng, on page 478
- Related topics, on page 478

RIPng

RIPng

RIPng uses User Datagram Protocol (UDP) data packets to exchange routing information in small internetworks.

RIPng supports IPv6 and uses the following two message types:

- Request—Sent to the multicast address FF02::9 to request route updates from other RIPng-enabled routers.
- Response—Sent every 30 seconds by default (see the Verify the RIPng configuration, on page 477 section). The router also sends response messages after it receives a request message. The response message contains the entire RIPng route table. RIPng sends multiple response packets for a request if the RIPng routing table cannot fit in one response packet.

RIPng uses a hop count for the routing metric. The hop count is the number of routers that a packet can traverse before reaching its destination. A directly connected network has a metric of 1. An unreachable network has a metric of 16. This small range of metrics makes RIPng an unsuitable routing protocol for large networks.

Split Horizon

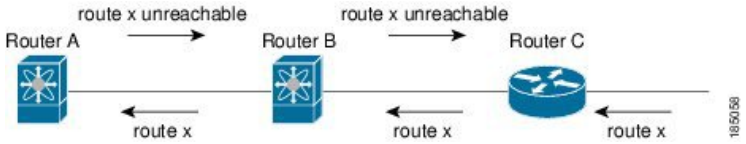
You can use split horizon to ensure that RIPng never advertises a route out of the interface where it was learned.

Split horizon is a method that controls the sending of RIPng update and query packets. When you enable split horizon on an interface, Cisco NX-OS does not send update packets for destinations that were learned from this interface. Controlling update packets in this manner reduces the possibility of routing loops.

You can use split horizon with poison reverse to configure an interface to advertise routes learned by RIPng as unreachable over the interface that learned the routes.

The following figure shows a sample RIPng network with split horizon and poison reverse enabled.

Figure 37: RIPng with Split Horizon Poison Reverse



Router C learns about route X and advertises that route to Router B. Router B in turn advertises route X to Router A but sends a route X unreachable update back to Router C.

By default, split horizon is enabled on all interfaces.

Route filter

You can configure a route policy on an RIPng-enabled interface to filter the RIPng updates. Cisco NX-OS updates the route table with only those routes that the route policy allows.

Load balance

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 usage of network segments and increases effective network bandwidth.

Cisco NX-OS supports the Equal Cost Multiple Paths (ECMP) feature with up to 16 equal-cost paths in the RIPng route table and the unicast RIB. You can configure RIPng to load balance traffic across some or all of those paths.

Default information origination and generation

Cisco NX-OS supports default-information origination and generation for RIPng IPv6.

To generate a default route into the Routing Information Protocol (RIP), use the default-information originate command in router address-family configuration mode. To disable this feature, use the **no** form of this command.

default-information originate [**always**] [**route-map** *map-name*]

no default-information originate

Note Use the always keyword to generate the default route if the route is not present in the RIP routing information base, that is the RIP internal RIB. Use the route-map keyword along with the map-name variable to generate the default route only if the route is permitted by the route map. The map name is any alphanumerical string up to 63 characters. Use the originate to send the default route along with regular updates.

The following example shows how to originate a default route to all routes that pass the condition route map.

switch(config)# **router rip Enterprise**

switch(config-router)# **address-family ipv6 unicast**

switch(config-router-af)# **default-information originate route-map Condition**

High availability for RIPng

Cisco NX-OS supports stateless restarts for RIPng. After a reboot or supervisor switchover, Cisco NX-OS applies the running configuration, and RIPng immediately sends request packets to repopulate its routing table.

Virtualization for RIPng

Cisco NX-OS supports multiple instances of the RIPng protocol that run on the same system. RIPng supports virtual routing and forwarding (VRF) instances.

Prerequisites for RIPng

RIPng has these prerequisites:

- You must enable RIPng (see the Enable RIPng, on page 468 section).

Guidelines and limitations for RIPng

RIPng has the following configuration guidelines and limitations:

- Beginning with Cisco NX-OS Release 10.2(3)F, RIPng feature is introduced to support IPv6 on Cisco Nexus 9300 and 9500 series platform switches.
- 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 not two different entries.
- Cisco NX-OS does not support RIPv1. If Cisco NX-OS receives an RIPv1 packet, it logs a message and drops the packet.
- Cisco NX-OS does not establish adjacencies with RIPv1 routers.

Default settings for RIPng parameters

The table lists the default settings for RIPng parameters.

Default RIPng Parameters

Parameters	Default
Maximum paths for load balancing	16
RIPng feature	Disabled
Split horizon	Enabled

Configure RIPng

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.

Enable RIPng

You must enable RIPng before you can configure RIPng.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 [**no**] **feature rip**

Example:

```
switch(config)# feature rip
```

Enables the RIPng 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.

Create an RIPng instance

You can create an RIPng instance and configure the address family for that instance.

Before you begin

You must enable RIPng (see the Enable RIPng, on page 468 section).

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **[no] router rip** *instance-tag*

Example:

```
switch(config)# router RIP Enterprise
      switch(config-router)#
```

Creates a new RIPng instance with the configured *instance-tag*.

Step 3 **address-family ipv6 unicast**

Example:

```
switch(config-router)# address-family ipv6 unicast
      switch(config-router-af)#
```

Configures the address family for this RIPng instance and enters address-family configuration mode.

Step 4 (Optional) **show ipv6 rip** [**instance** *instance-tag*] [**vrf** *vrf-name*]

Example:

```
switch(config-router-af)# show ipv6 rip
```

Displays a summary of RIPng information for all RIPng instances.

Step 5 (Optional) **distance** *value*

Example:

```
switch(config-router-af)# distance 30
```

Sets the administrative distance for RIPng. The range is from 1 to 255. The default is 120. See the Administrative Distance section.

Step 6 (Optional) **maximum-paths** *number*

Example:

```
switch(config-router-af)# maximum-paths 6
```

Configures the maximum number of equal-cost paths that RIPng maintains in the route table. The range is from 1 to 64. The default is 16.

Step 7 (Optional) **copy running-config startup-config**

Example:

```
switch(config-router-af)# copy running-config
      startup-config
```

Saves this configuration change.

This example shows how to create an RIPng instance for IPv6 and set the number of equal-cost paths for load balancing:

```
switch# configure terminal
    switch(config)# router rip Enterprise
    switch(config-router)# address-family ipv6 unicast
    switch(config-router-af)# max-paths 10
    switch(config-router-af)# copy running-config startup-config
```

Restart an RIPng instance

You can restart an RIPng instance and remove all associated neighbors for the instance.

To restart an RIPng instance and remove all associated neighbors, use the following command in global configuration mode:

Procedure

---

restart rip *instance-tag*

Example:

```
switch(config)# restart rip Enterprise
```

Restarts the RIPng instance and removes all neighbors.

Configure RIPng on an interface

Before you begin

You must enable RIPng (see the Enable RIPng, on page 468 section).

Procedure

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 **ipv6 router rip** *instance-tag*

Example:

```
switch(config-if)# ipv6 router rip Enterprise
```

Associates this interface with an RIPng instance.

Step 4 (Optional) **show ipv6 rip** [**instance** *instance-tag*] **interface** [*interface-type slot/port*] [**vrf** *vrf-name*] [**detail**]

Example:

```
switch(config-if)# show ipv6 rip Enterprise ethernet 1/2
```

Displays RIPng information for an interface.

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

Example:

```
switch(config-if)# copy running-config startup-config
```

Saves this configuration change.

This example shows how to add Ethernet 1/2 interface to an RIPng instance:

```
switch# configure terminal
    switch(config)# interface ethernet 1/2
    switch(config-if)# ipv6 router rip Enterprise
    switch(config)# copy running-config startup-config
```

Configure split horizon with poison reverse

You can configure an interface to advertise routes learned by RIPng as unreachable over the interface that learned the routes by enabling poison reverse.

To configure split horizon with poison reverse on an interface, use the following command in interface configuration mode:

Procedure

ipv6 rip poison-reverse

Example:

```
switch(config-if)# ipv6 rip poison-reverse
```

Enables split horizon with poison reverse. Split horizon with poison reverse is disabled by default.

Configure Cisco NX-OS RIPng for compatibility with Cisco IOS RIPng

You can configure Cisco NX-OS RIPng to behave like Cisco IOS RIPng in the way that routes are advertised and processed.

Directly connected routes are treated with cost 1 in Cisco NX-OS RIPng and with cost 0 in Cisco IOS RIPng. When routes are advertised in Cisco NX-OS RIPng, the receiving device adds a minimum cost of +1 to all received routes and installs the routes in its routing table. In Cisco IOS RIPng, this cost increment is done on the sending router, and the receiving router installs the routes without any modification. This difference in behavior can cause issues when both Cisco NX-OS and Cisco IOS devices are working together. You can prevent these compatibility issues by configuring Cisco NX-OS RIPng to advertise and process routes like Cisco IOS RIPng.

Before you begin

You must enable RIPng (see the Enable RIPng, on page 468 section).

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **router rip** *instance-tag*

Example:

```
switch(config)# router rip 100
      switch(config-router)#
```

Creates a new RIPng instance with the configured instance tag. You can enter 100, 201, or up to 20 alphanumeric chapters for the instance tag.

Step 3 [**no**] **metric direct 0**

Example:

```
switch(config-router)# metric direct 0
```

Configures all directly connected routes with cost 0 instead of the default of cost 1 in order to make Cisco NX-OS RIPng compatible with Cisco IOS RIPng in the way that routes are advertised and processed.

Note This command must be configured on all Cisco NX-OS devices that are present in any RIPng network that also contains Cisco IOS devices.

Step 4 (Optional) **show running-config rip**

Example:

```
switch(config-router)# show
      running-config rip
```

Displays the current running RIPng configuration.

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

Example:

```
switch(config-router)# copy running-config
      startup-config
```

Saves this configuration change.

This example shows how to disable Cisco NX-OS RIPng compatibility with Cisco IOS RIPng by returning all direct routes from cost 0 to cost 1:

```
switch# configure terminal
    switch(config)# router rip 100
    switch(config-router)# no metric direct 0
    switch(config-router)# show running-config rip
    switch(config-router)# copy running-config startup-config
```

Configure virtualization

You can configure multiple RIPng instances, create multiple VRFs, and use the same or multiple RIPng instances in each VRF. You assign an RIPng 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 configurations for that interface.

Before you begin

You must enable RIPng (see the Enable RIPng, on page 468 section).

Procedure

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 **exit**

Example:

```
switch(config-vrf)# exit
      switch(config)#
```

Exits VRF configuration mode.

Step 4 **router rip** *instance-tag*

Example:

```
switch(config)# router rip Enterprise
      switch(config-router)#
```

Creates a new RIPng instance with the configured instance tag.

Step 5 **vrf** *vrf-name*

Example:

```
switch(config-router)# vrf RemoteOfficeVRF
      switch(config-router-vrf)#
```

Creates a new VRF.

Step 6 (Optional) **address-family ipv6 unicast**

Example:

```
switch(config-router-vrf)# address-family ipv6 unicast
      switch(config-router-vrf-af)#
```

Configures the VRF address family for this RIPng instance.

Step 7 **interface ethernet** *slot*/*port*

Example:

```
switch(config-router-vrf-af)# interface ethernet 1/2
      switch(config-if)#
```

Enters interface configuration mode.

Step 8 **vrf member** *vrf-name*

Example:

```
switch(config-if)# vrf member RemoteOfficeVRF
```

Adds this interface to a VRF.

Step 9 **ipv6 address** *ipv6-prefix*/*length*

Example:

```
switch(config-if)# ipv6 address 1001::1/64
```

Configures an IP address for this interface. You must perform this step after you assign this interface to a VRF.

Step 10 **ipv6 router rip** *instance-tag*

Example:

```
switch(config-if)# ipv6 router rip Enterprise
```

Associates this interface with an RIPng instance.

Step 11 (Optional) **show ipv6 rip** [**instance** *instance-tag*] **interface** [*interface-type slot*/*port*] [**vrf** *vrf-name*]

Example:

```
switch(config-if)# show ipv6 rip Enterprise ethernet 1/2
```

Displays RIPng information for an interface in a VRF.

Step 12 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if)# copy running-config startup-config
```

Saves this configuration change.

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 rip Enterprise
    switch(config-router)# vrf RemoteOfficeVRF
    switch(config-router-vrf)# address-family ipv6 unicast
    switch(config-router-vrf-af)# interface ethernet 1/2
    switch(config-if)# vrf member RemoteOfficeVRF
    switch(config-if)# ipv6 address 1001::1/64
    switch(config-if)# ipv6 router rip Enterprise
    switch(config-if)# copy running-config startup-config
```

Tune RIPng

You can tune RIPng to match your network requirements. RIPng uses several timers that determine the frequency of routing updates, the length of time before a route becomes invalid, and other parameters. You can adjust these timers to tune routing protocol performance to better suit your internetwork needs.

Note You must configure the same values for the RIPng timers on all RIPng-enabled routers in your network.

You can use the following optional commands in address-family configuration mode to tune RIPng:

Command	Purpose
timers basic *update timeout holddown garbage-collection* **Example:** `switch(config-router-af)# timers basic 40 120 120 100`	Sets the RIPng timers in seconds. The parameters are as follows: • *update*—The range is from 5 to any positive integer. The default is 30. • *timeout*—The time that Cisco NX-OS waits before declaring a route as invalid. If Cisco NX-OS does not receive route update information for this route before the timeout interval ends, Cisco NX-OS declares the route as invalid. The range is from 1 to any positive integer. The default is 180. • *holddown*—The time during which Cisco NX-OS ignores better route information for an invalid route. The range is from 0 to any positive integer. The default is 180. • *garbage-collection*—The time from when Cisco NX-OS marks a route as invalid until Cisco NX-OS removes the route from the routing table. The range is from 1 to any positive integer. The default is 120.

You can use the following optional commands in interface configuration mode to tune RIPng:

Command	Purpose
ipv6 rip route-filter {**prefix-list** *list-name* | **route-map** *map-name* | [**in** | **out**] **Example:** `switch(config-if)# ipv6 rip route-filter route-map InputMap in`	Specifies a route map to filter incoming or outgoing RIPng updates.

Verify the RIPng configuration

To display the RIPng configuration, perform one of the following tasks.

Command	Purpose
show ipv6 rip instance [*instance-tag*] [**vrf** *vrf-name*]	Displays the status for an instance of RIPng.
show ipv6 rip [**instance** *instance-tag*] **interface** *slot/port* **detail** [**vrf** *vrf-name*]	Displays the RIPng status for an interface.
show ipv6 rip [**instance** *instance-tag*] **neighbor** [*interface-type number*] [**vrf** *vrf-name*]	Displays the RIPng neighbor table.

Command	Purpose
show ipv6 rip [**instance** *instance-tag*] **route** [*ip-prefix/length* [**longer-prefixes** | **shorter-prefixes**]] [**summary**] [**vrf** *vrf-name*]	Displays the RIPng route table.
show running-configuration rip	Displays the current running RIPng configuration.

Display RIPng statistics

To display RIPng statistics, use the following commands.

Command	Purpose
show ipv6 rip [**instance** *instance-tag*] **statistics** *interface-type number* [**vrf** *vrf-name*]	Displays the RIPng statistics.

Use the **clear ipv6 rip statistics** command to clear RIPng statistics.

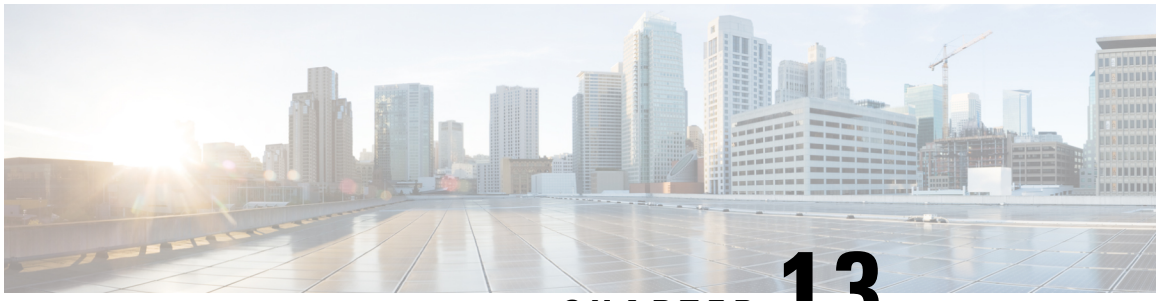
Configuration Examples for RIPng

The following example shows how to create the Enterprise RIPng instance in a VRF and add Ethernet interface 1/2 to this RIPng instance.

```
router rip Enterprise
address-family ipv6 unicast
distance 33
maximum-paths 8
default-information originate always
timers basic 31 181 181 121
interface ethernet 1/2
ipv6 router rip Enterprise
```

Related topics

See Configuring Route Policy Manager, on page 523 for more information on route maps.



CHAPTER 13

Configure static route

This chapter describes how to configure static routing on the Cisco NX-OS device.

This chapter contains these sections:

- About static route, on page 479
- Prerequisites for static routing, on page 481
- Default settings, on page 481
- Configuring static routing, on page 481
- Configuration example for static route, on page 486

About static route

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 might 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 must 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 that 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, but only for point-to-point interfaces. For broadcast interfaces, the next hop must be an IPv4/IPv6 address.

Fully specified static routes

You must specify either the output interface (the interface on which all packets are sent to the destination network) or 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 hops that have reachability to the remote next hops.

BFD

This feature supports bidirectional forwarding detection (BFD). BFD is a detection protocol that is 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 9000 Series NX-OS Interfaces Configuration Guide, Release 9.3(x) for more information. .

Virtualization support

Static routes support virtual routing and forwarding (VRF) instances.

Prerequisites for static routing

Static routing has these prerequisites:

- A static route is not added to the unicast routing table if there is no unicast route containing its next hop address. This check does not apply to static routes where the next hop is in a directly connected subnet.
- If the next-hop for a static route is resolved by a directly connected route, it is expected that the static route will be installed in the URIB (routing table). For tracking next-hop reachability of such static routes, adding BFD to the static route or using Object Tracking to verify reachability of the next-hop is recommended to avoid black-holing traffic to an unreachable next-hop destination.

Default settings

The table lists the default settings for static routing parameters.

Table 32: Default Static Routing Parameters

Parameters	Default
Administrative distance	1
RIP feature	Disabled

Configuring static routing

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.

Note Beginning with Cisco NX-OS Release 10.6(2)F, static routing is supported on Cisco Nexus N9324C-SE1U and N9348Y2C6D-SE1U platform switches.

Configure a static route

You can configure a static route on the device.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 Enter one of these commands:

- **ip route** {*ip-prefix* | *ip-addr*/*ip-mask*} {[*next-hop* | *nh-prefix*] | [*interface next-hop* | *nh-prefix*]} [**name** *nexthop-name*] [**tag** *tag-value*] [*preference*]
- **ipv6 route** *ipv6-prefix* {*nh-prefix* | *link-local-nh-prefix*} | {*nexthop* [*interface*] | *link-local-nexthop* [*interface*]} [**name** *nexthop-name*] [**tag** *tag-value*] [*preference*]

Example:

```
switch(config)# ip route 192.0.2.0/8 ethernet 1/2 192.0.2.4

      switch(config)# ipv6 route 2001:0DB8::/48 6::6 ethernet 2/1
```

Configures a static route and the interface for this static route. Use **?** to display a list of supported interfaces. You can specify a null interface by using **null 0**.

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.

Note Use the **no** {**ip** | **ipv6**} **route** command to remove the static route.

Step 3 (Optional) **show** {**ip** | **ipv6**} **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.

This example shows how to configure a static route for a null interface:

```
switch# configure terminal
    switch(config)# ip route 1.1.1.1/32 null 0
    switch(config)# copy running-config startup-config
```

Configure a static route over a VLAN

You can configure a static route without next-hop support over a VLAN.

Before you begin

Ensure that the access port is part of the VLAN.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **feature interface vlan**

Example:

```
switch(config)# feature interface-vlan
```

Enables VLAN interface mode.

Step 3 **interface-vlan** *vlan-id*

Example:

```
switch(config)# interface-vlan 10
```

Creates an SVI and enters interface configuration mode.

The range for **vlan-id** argument is from 1 to 4094, except for the VLANs reserved for the internal switch.

Step 4 **ip address** *ip-addr*/*length*

Example:

```
switch(config)# ip address 192.0.2.1/8
```

Configures an IP address for the VLAN.

Step 5 [**no**] **ip route** *ip-addr*/*length vlan-id*

Example:

```
switch(config)# ip route
      209.165.200.224/27 vlan 10
```

Adds an interface static route without a next hop on the switch virtual interface (SVI).

The IP address is the address that is configured on the interface that is connected to the switch.

Use the **no** keyword with this command to remove the static route.

Step 6 (Optional) **show ip route**

Example:

```
switch(config)# show ip route
```

Displays routes from the Unicast Route Information Base (URIB).

Step 7 (Optional) **copy running-config startup-config**

Example:

```
switch(config)# copy running-config
      startup-config
```

Saves this configuration change.

This example shows how to configure a static route without a next hop over an SVI:

```
switch# configure terminal
    switch(config)# feature interface-vlan
    swicth(config)# interface vlan 10
    switch(config-if)# ip address 192.0.2.1/8
    switch(config-if)# ip route 209.165.200.224/27 vlan 10  <===209,165.200.224 is the IP
    address of the interface that is configured on the interface that is directly connected
 to
    the switch.
    switch(config-if)# copy running-config startup-config
```

Configure virtualization

You can configure a static route in a VRF.

Note When a **ip route** command is applied on a VRF context, the **show run vrf** command displays some octets that have changed from the initial configuration.

Procedure

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 StaticVrf
      switch(config-vrf)#
```

Creates a VRF and enters VRF configuration mode.

Step 3 Enter one of these commands:

- **ip route** {*ip-prefix* | *ip-addr ip-mask*} {*next-hop* | *nh-prefix* | *interface*} [**name** *nexthop-name*] [**tag** *tag-value*] [*preference*]
- **ipv6 route** *ipv6-prefix* {*nh-prefix* | *link-local-nh-prefix*} | {*nexthop* [*interface*] | *link-local-nexthop* [*interface*]} [**name** *nexthop-name*] [**tag** *tag-value*] [*preference*]

Example:

```
switch(config-vrf)# ip route 192.0.2.0/8
      ethernet 1/2


switch(config-vrf)# ipv6 route
      2001:0DB8::/48 6::6 ethernet 2/1
```

Configures a static route and the interface for this static route. Use **?** to display a list of supported interfaces. You can specify a null interface by using **null 0**.

You can optionally configure the next-hop address.

The *preference* value sets the administrative distance. The range is from 1 through 255. The default is 1.

Step 4 (Optional) **show** {**ip** | **ipv6**} **static-route vrf** *vrf-name*

Example:

```
switch(config-vrf)# show ip static-route
```

Displays information about static routes.

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

Example:

```
switch(config-vrf)# copy running-config
      startup-config
```

Saves this configuration change.

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
```

Verify the static routing configuration

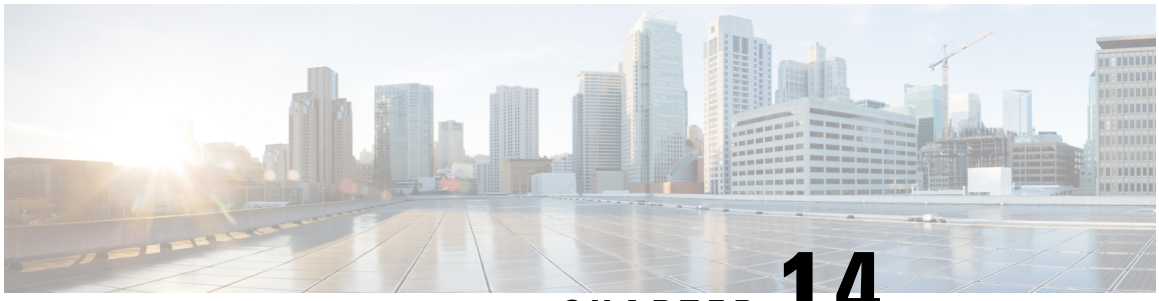
To display the static routing configuration, perform one of the following tasks:

Command	Purpose
show {**ip** | **ipv6**} **static-route**	Displays the configured static routes.
show ipv6 static-route vrf *vrf-name*	Displays static route information for each VRF.
show {**ip** | **ipv6**} **static-route track-table**	Displays information about the IPv4 or IPv6 static-route track table.

Configuration example for static route

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 14

Configure Layer 3 virtualization

Layer 3 virtualization is a configuration process on the Cisco NX-OS device.

- Layer 3 virtualization, on page 487
- Prerequisites for VRF, on page 492
- Guidelines and limitations for VRFs, on page 492
- Guidelines and limitations for VRF route leaking, on page 493
- Default settings, on page 493
- Configure VRFs, on page 494
- Verify the VRF configuration, on page 500
- Configuration examples for VRFs, on page 500
- Additional references, on page 508

Layer 3 virtualization

Layer 3 virtualization is a technology that enables multiple virtual routing and forwarding instances (VRFs) on a single device, each with its own address space and routing tables.

- Each VRF contains separate unicast and multicast route tables for IPv4 and IPv6.
- Routing decisions are made independently for each VRF.
- Each router has a default VRF and a management VRF.

VRF Types and Characteristics

The following sections describe the management VRF, default VRF, and the egress loadbalance resolution VRF.

- **Management VRF**: Used exclusively for management purposes. Only the mgmt 0 interface can be in the management VRF, and it cannot be assigned to another VRF. No routing protocols can run in the management VRF (static only).
- **Default VRF**: All Layer 3 interfaces exist in the default VRF until assigned to another VRF. Routing protocols run in the default VRF context unless another VRF context is specified. The default VRF uses the default routing context for all show commands and is similar to the global routing table concept in Cisco IOS.

- **Egress Loadbalance Resolution VRF**: An internal VRF created automatically to assist in additional computation and resolution of routes for the VXLAN EVPN feature. This VRF is not configurable by the user and cannot be deleted. The VRF limit is increased from 4096 to 4097 to accommodate this new implicit VRF.

Note When you upgrade to Cisco NX-OS Release 10.4(3)F, the default configuration for limit-resource will be changed to 4096.

When you upgrade to Cisco NX-OS Release 10.3(5), the default configuration for limit-resource will be changed to 4096.

Note

- The Egress Loadbalance Resolution VRF is not configurable by the user and cannot be deleted, as this is a benign empty VRF.
- The VRF limit is increased from 4096 to 4097 to accommodate this new implicit VRF.

  For example:

  - Existing default configuration

```
vdc switch id 1
        limit-resource vrf minimum 2 maximum 4096
```

  - New default configuration

```
vdc switch id 1
        limit-resource vrf minimum 2 maximum 4097
```

VRF and routing

- All unicast and multicast routing protocols support VRFs.
- Routing parameters configured in a VRF are independent from those in another VRF for the same routing protocol instance.
- Interfaces and route protocols can be assigned to a VRF to create virtual Layer 3 networks.

VRF and Routing Reference Information

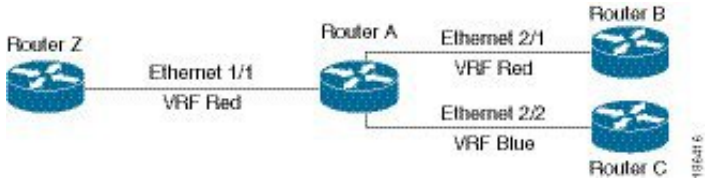
Each interface exists in only one VRF. A physical network can be split into multiple virtual networks using VRFs. Routers within the same VRF share route updates, while routers in different VRFs do not share route updates by default.

- Routers Z, A, and B exist in VRF Red and form one address domain.
- Router C is configured in a different VRF and does not share route updates with routers in VRF Red.

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.

- Cisco NX-OS supports route leaking (import or export) between VRFs.

Figure 38: VRFs in a Network



Route leaking and importing routes from the default VRF

Route leaking and importing routes from the default VRF is a process that allows the transfer of IP prefixes between VRFs in Cisco NX-OS using import policies and route maps.

- Import policies use route maps to specify which prefixes are imported into a VRF.
- Both IPv4 and IPv6 unicast prefixes can be imported.
- Imported prefixes cannot be reimported into another VRF through this policy.

Importing routes from the default VRF: details and considerations

To import IP prefixes from the default VRF, define match criteria in the import route map using standard route policy filtering mechanisms such as prefix lists or as-path filters.

- You can create an IP prefix list or an as-path filter to define an IP prefix or range, and use that in a match clause for the route map.
- Prefixes that pass through the route map are imported into the specified VRF using the import policy.

Note Routes in the BGP default VRF can be imported directly. Any other routes in the default VRF should be redistributed into BGP first.

For more information, see the Guidelines and Limitations for VRF Route Leaking section.

BGP VRF router-ID for IPv6 only environments

The following are the sources to obtain router-id in order of priority:

1. VRF level router-id command
2. IPv4 address configured VRF interface
3. Inherit non-default VRF router-id from default VRF router-id config

Note The third source for router-id has the least priority and applies only if the first and second sources are unavailable.

Note In the absence of router-id, BGP OPEN messages cannot be sent.

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—See the Cisco Nexus 9000 Series NX-OS Security Configuration Guide for more information.
- Call Home—See the Cisco Nexus 9000 Series NX-OS System Management Configuration Guide for more information.
- DNS—See DNS, on page 103 for more information.
- HSRP—See Configuring HSRP, on page 609 for more information.
- HTTP—See the Cisco Nexus 9000 Series NX-OS Fundamentals Configuration Guide for more information.
- NTP—See the Cisco Nexus 9000 Series NX-OS System Management Configuration Guide for more information.
- Ping and Traceroute—See the Cisco Nexus 9000 Series NX-OS Fundamentals Configuration Guide for more information.
- RADIUS—See the Cisco Nexus 9000 Series NX-OS Security Configuration Guide for more information.
- SNMP—See the Cisco Nexus 9000 Series NX-OS System Management Configuration Guide for more information.
- SSH—See the Cisco Nexus 9000 Series NX-OS Security Configuration Guide for more information.
- Syslog—See the Cisco Nexus 9000 Series NX-OS System Management Configuration Guide for more information.
- TACACS+—See the Cisco Nexus 9000 Series NX-OS Security Configuration Guide for more information.
- TFTP—See the Cisco Nexus 9000 Series NX-OS Fundamentals Configuration Guide for more information.
- VRRP—See Configure VRRP, on page 643 for more information.
- XML—See the Cisco NX-OS XML Management Interface User Guide for more information.

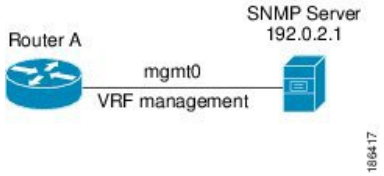
See the appropriate configuration guide for each service for more information on configuring VRF support in that service.

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 Cisco NX-OS must use to reach the server.

The following figure 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.

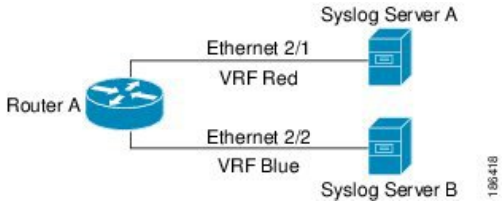
Figure 39: Service VRF Reachability



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 40: Service VRF Filtering

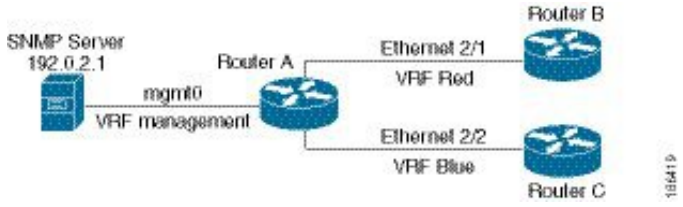


Combine reachability and filtering

You can combine reachability and filtering for VRF-aware services. You can 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 41: Service VRF Reachability Filtering



Prerequisites for VRF

You must install the Advanced Services license to use virtual device contexts (VDCs) besides the default VDC. The license requirement for VRF is same as VDC.

Guidelines and limitations for VRFs

VRFs have the following configuration guidelines and limitations:

- 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 not two different entries.
- When you make an interface a member of an existing VRF, Cisco NX-OS removes all Layer 3 configurations. 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 configurations. You must use the **write erase** command and then the **write erase boot** command.
- The following guidelines and limitations are for route targets:
  - It is a best practice to assign different route targets for Layer-2 and Layer-3.
  - For automatic route-target generation, route targets are generated from their EVIs. It is a best practice to have different EVI ranges for Layer 2 and Layer 3, which ensures that Layer-2 and Layer-3 EVIs do not use the same identifier.
- Beginning with Cisco NX-OS Release 10.3(1)F , multi VRF is supported on the Cisco Nexus 9808 platform switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, multi VRF is supported on the Cisco Nexus 9804 platform switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, multi VRF is supported on the Cisco Nexus X98900CD-A and N9KX9836DM-A line cards with Cisco Nexus 9808 and 9804 switches.
- Beginning with Cisco NX-OS Release 10.4(2)F, multi VRF is supported on the Cisco Nexus N9K-C9232E-B1 platform switches.
- Beginning with Cisco NX-OS Release 10.6(2)F, VRF is supported on Cisco Nexus N9324C-SE1U and N9348Y2C6D-SE1U platform switches.

Guidelines and limitations for VRF route leaking

VRF route leaking has these configuration guidelines and limitations:

- Route leaking is supported between any two non-default VRFs and from the default VRF to a non-default VRF.

Note Route leaking between VRFs is not supported for MPLS Segment Routing (SR-MPLS).

Route leaking between VRFs is not supported for BGP. A BGP speaker cannot connect to a peer IP that is routed through a different VRF.

- You can restrict route leaking to specific routes using route map filters to match designated IP addresses.
- Route synchronization between NX-OS and Guestshell container does not happen when the route points towards another VRF.
- By default, the maximum number of IP prefixes that can be imported from the default VRF into a non-default VRF and vice versa is 1000 routes.
- You can modify the default route leak value of 1000 routes that are imported from default to non-default VRF using these commands:

```
vrf context RED
  address-family ipv4 unicast
    import vrf default <prefix-limit> map test
```

- There is no limit on the number of routes that can be leaked between two non-default VRFs.
- Beginning with Cisco NX-OS Release 10.3(1)F , route leak between VRFs is supported on the Cisco Nexus 9808 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, route leak between VRFs is supported on the Cisco Nexus 9804 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, route leak between VRFs is supported on Cisco Nexus X98900CD-A and X9836DM-A line cards with 9808 and 9804 switches.
- Beginning with Cisco NX-OS Release 10.4(2)F, route leak between VRFs is supported on Cisco Nexus 9232E-B1 switches.

Default settings

The table lists the default settings for VRF parameters.

Table 33: Default VRF Parameters

Parameters	Default
Configured VRFs	Default, management
Routing context	Default VRF

Configure VRFs

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.

Create VRF

You can create a VRF.

Note Any commands available in global configuration mode are also available in VRF configuration mode.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 [**no**] **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.

Using the **no** option with this command deletes the VRF and all associated configurations.

Step 3 (Optional) **ip route** {*ip-prefix* | *ip-addr ip-mask*} {[*next-hop* | *nh-prefix*] | [*interface next-hop* | *nh-prefix*]} [**tag** *tag-value* [*preference*]

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.

This example show 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
```

Assign VRF membership to an interface

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 **interface***interface-typeslot/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 (Optional) **show vrf***vrf-name***interface***interface-typenumber*

Example:

```
switch(config-vrf)# show vrf Enterprise
      interface ethernet 1/2
```

Displays VRF information.

Step 6 (Optional) **copy running-config startup-config**

Example:

```
switch(config-vrf)# copy running-config
      startup-config
```

Saves this configuration change.

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
```

Configure 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

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

Step 2 **router ospf** *instance-tag*

Example:

```
switch (config-vrf)# router ospf 201
      switch(config-router)#
```

Creates a new OSFPv2 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. This command is used for load balancing.

Step 5 **exit**

Example:

```
switch(config-router-vrf)# exit
      switch(config-router)#
```

Exits VRF configuration mode.

Step 6 **exit**

Example:

```
switch(config-router)# exit
      switch(config)#
```

Exits router configuration mode.

Step 7 **interface** *interface-type slot/port*

Example:

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

Enters interface configuration mode.

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-if)# copy running-config
      startup-config
```

Copies the running configuration to the startup configuration.

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
```

Configure a VRF-Aware service

You can configure a VRF-aware service for reachability and filtering.

This section uses SNMP and IP domain lists as example services for the detailed configuration steps.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

```
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.

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 context Blue
    switch(config-vrf)# snmp-server host 192.0.2.12 use-vrf Red
    switch(config)# copy running-config startup-config
```

Set the VRF scope

You can set the VRF scope for all EXEC commands (for example, **show** commands). Doing so 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.

Procedure

---

routing-context vrf *vrf-name*

Example:

```
switch# routing-context vrf red
      switch%red#
```

Sets the routing context for all EXEC commands. The default routing context is the default VRF.

Note Use the **routing-context vrf default** command to return to the default VRF scope.

To return to the default VRF scope, use the following command in EXEC mode:

Command	Purpose

routing-context vrf default
Example:
`switch%red# routing-context vrf default`
`switch#`
	Sets the default routing context.

Note When you do a shutdown of the VPN VRF with BGP configurations, it will take around 50 seconds for the shutdown process to complete.

Verify the VRF configuration

To display VRF configuration information, perform one of the following tasks.

Command	Purpose
show bgp process vrf [*vrf-name*]	Displays the information for all or one VRF.
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

For more information on VRF configuration, see Configure VRFs, on page 494.

Note

- The **snmp-server host** command is available both globally and under the **vrf-context** command. The following example shows the configuration under **vrf-context**.
- Ensure that the host is configured on the switch before attaching any attributes (such as use-vrf, source-interface, filter-vrf etc).

Configuration Example for VRF Red

```
!SNMP server configuration under VRF context Red:
vrf context Red
  snmp-server host 192.168.0.12 use-vrf Red

!OSPF instance configuration to VRF Red
router ospf 201
  vrf Red

!interface configuration for VRF Red
interface ethernet 1/2
  vrf member Red
  ip address 192.168.0.1/16
  ip router ospf 201 area 0
  no shutdown
```

Configuration Example for VRF Red and Blue

```
!VRFs (Red, and Blue) creation
vrf context Red
vrf context Blue

!Configures OSPF per VRF
feature ospf
router ospf Lab
  vrf Red

router ospf Production
  vrf Blue
      router-id 192.168.1.0

interface ethernet 1/2
  vrf member Red
  ip address 192.168.0.1/16
  ip router ospf Lab area 0
  no shutdown

interface ethernet 10/2
  vrf member Blue
  ip address 192.168.0.1/16
  ip router ospf Production area 0
  no shutdown

!SNMP server configuration under VRF
!Note: Use the SNMP context "lab" to access the OSPF-MIB values for the OSPF instance Lab
in VRF "Red" in this example.
!Create SNMP entities (v2c and/or v3) with appropriate groups/roles as needed on the switch
 to access the MIBs on the switch
!Create SNMP v3 user that can be used for SNMP queries, for example:
snmp-server user admin network-admin auth md5 password1
!Create SNMP v2c community  that can be used for SNMP queries, for example:
snmp-server community public ro
```

```
!Create SNMP contexts that can be used along with the entities for SNMP queries, for example:
snmp-server context lab instance Lab vrf Red
snmp-server context production instance Production vrf Blue
```

VRFs Configuration Example for Route Leaking

```
!VRF configuration
feature bgp
vrf context red
  ip route 192.168.33.0/32 192.168.3.1
  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 red
  ip address 192.168.3.2/24
  no shutdown

vrf context blue
  ip route 192.168.44.0/32 192.168.4.1
  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 blue
  ip address 192.168.4.2/24
  no shutdown
!IP prefix list configuration
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 192.168.101.101/32 192.168.55.1
!BGP per VRF assignment
router bgp 100
  address-family ipv4 unicast
    redistribute static route-map test

  vrf red
    address-family ipv4 unicast
      redistribute static route-map test

 vrf blue
    address-family ipv4 unicast
      redistribute static route-map test
```

Verification Example for route leaking between global and non-default VRFs

```
switch# 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>
```

```
192.168.55.0/24, ubest/mbest: 1/0, attached
    *via 192.168.55.5, Lo0, [0/0], 00:07:59, direct
192.168.55.5/32, ubest/mbest: 1/0, attached
    *via 192.168.55.5, Lo0, [0/0], 00:07:59, local
192.168.101.101/32, ubest/mbest: 1/0
    *via 192.168.55.1, [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 "red"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

192.168.33.0/32, ubest/mbest: 1/0
    *via 192.168.3.1, [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
192.168.44.0/32, ubest/mbest: 1/0
    *via 192.168.4.1%blue, [20/0], 00:12:08, bgp-100, external, tag 100
192.168.101.101/32, ubest/mbest: 1/0
    *via 192.168.55.1%default, [20/0], 00:07:41, bgp-100, external, tag 100
!
IP Route Table for VRF "blue"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>
192.168.33.0/32, ubest/mbest: 1/0
    *via 192.168.3.1%red, [20/0], 00:12:34, bgp-100, external, tag 100
192.168.44.0/32, ubest/mbest: 1/0
    *via 192.168.4.1, [1/0], 00:23:16, static
45.45.1.0/24, ubest/mbest: 1/0, attached
    *via 192.168.4.2, Eth1/11, [0/0], 00:25:53, direct
192.168.4.2/32, ubest/mbest: 1/0, attached
    *via 192.168.4.2, Eth1/11, [0/0], 00:25:53, local
192.168.101.101/32, ubest/mbest: 1/0
    *via 192.168.55.1%default, [20/0], 00:07:41, bgp-100, external, tag 100
switch(config)#
```

Configuration Example of Export VRF Default

The following example shows how to allow re-importation of already imported routes that is introduced in the “export vrf default” command to allow VPN imported routes to be re-imported into the default-VRF.

```
vrf context vpn1
  address-family ipv4 unicast
    export vrf default [<prefix-limit>] map <route-map> [allow-vpn]
  address-family ipv6 unicast
    export vrf default [<prefix-limit>] map <route-map> [allow-vpn]
```

Configuration Example of Border-leaf Configuration

- To configure IP prefix list, use the following commands:

```
!IP prefix list configuration
ip prefix-list DEFAULT_ROUTE seq 5 permit 0.0.0.0/0
route-map NO_DEFAULT_ROUTE deny 5
  match ip address prefix-list DEFAULT_ROUTE
route-map NO_DEFAULT_ROUTE permit 10
route-map allow permit 10

!Creation of VRFs, and importing the route maps
vrf context vni100
  vni 100
  ip route 0.0.0.0/0 Null0
  rd auto
  address-family ipv4 unicast
    route-target import 100:200
    route-target import 100:200 evpn
    route-target both auto
    route-target both auto evpn
    import vrf default map allow
    export vrf default map NO_DEFAULT_ROUTE allow-vpn

vrf context vni200
  vni 200
  ip route 0.0.0.0/0 Null0
  rd auto
  address-family ipv4 unicast
    route-target import 100:100
    route-target import 100:100 evpn
    route-target both auto
    route-target both auto evpn
    import vrf default map allow
    export vrf default map NO_DEFAULT_ROUTE

!BGP configuration
router bgp 100
  address-family ipv4 unicast
    redistribute direct route-map allow
  address-family ipv6 unicast
    redistribute direct route-map allow

  neighbor 192.168.101.101
    remote-as 100
    update-source loopback0
    address-family l2vpn evpn
      send-community extended

  neighbor 192.168.30.2
    remote-as 300
    address-family ipv4 unicast

  vrf vni100
    address-family ipv4 unicast
      network 0.0.0.0/0
      advertise l2vpn evpn
      redistribute direct route-map allow

  vrf vni200
    address-family ipv4 unicast
      network 0.0.0.0/0
      advertise l2vpn evpn
      redistribute direct route-map allow
```

Verification Example of BGP IPv4 Unicast configuration

```
switch(config-vrf)# show bgp ipv4 unicast 192.168.11.11/32
BGP routing table information for VRF default, address family IPv4 Unicast
BGP routing table entry for 192.168.11.11/32, version 14
Paths: (1 available, best #1)
Flags: (0x08041a) on xmit-list, is in urib, is best urib route, is in HW

  Advertised path-id 1
  Path type: internal, path is valid, is best path, in rib
             Imported from 192.168.3.3:3:192.168.11.11/32 (VRF vni100)
  AS-Path: 150 , path sourced external to AS
    192.168.1.0 (metric 81) from 192.168.101.101 (192.168.101.101)
      Origin incomplete, MED 0, localpref 100, weight 0
      Received label 100
      Extcommunity:
          RT:100:100
          ENCAP:8
          Router MAC:5254.004e.a437
      Originator: 192.168.1.0 Cluster list: 192.168.101.101

  Path-id 1 advertised to peers:
    192.168.30.2
```

Verification Example of BGP IPv4 Unicast configuration per VRF

```
switch(config-vrf)# show bgp vrf vni100 ipv4 unicast 192.168.11.11/32
BGP routing table information for VRF vni100, address family IPv4 Unicast
BGP routing table entry for 192.168.11.11/32, version 8
Paths: (1 available, best #1)
Flags: (0x08041e) on xmit-list, is in urib, is best urib route, is in HW
  vpn: version 19, (0x100002) on xmit-list

  Advertised path-id 1, VPN AF advertised path-id 1
  Path type: internal, path is valid, is best path, in rib
             Imported from 192.168.1.0:3:[5]:[0]:[0]:[32]:[192.168.11.11]:[0.0.0.0]/224
  AS-Path: 150 , path sourced external to AS
    192.168.1.0 (metric 81) from 192.168.101.101 (192.168.101.101)
      Origin incomplete, MED 0, localpref 100, weight 0
      Received label 100
      Extcommunity:
          RT:100:100
          ENCAP:8
          Router MAC:5254.004e.a437
      Originator: 192.168.1.0 Cluster list: 192.168.101.101

  VRF advertise information:
  Path-id 1 not advertised to any peer

  VPN AF advertise information:
  Path-id 1 not advertised to any peer
```

Verification Examples of BGP IPv6 Unicast configuration

```
switch(config-vrf)# show bgp ipv6 unicast 2001:DB8:1::1/64
BGP routing table information for VRF default, address family IPv6 Unicast
BGP routing table entry for 2001:DB8:1::1/64, version 13
Paths: (1 available, best #1)
Flags: (0x08041a) on xmit-list, is in u6rib, is best u6rib route, is in HW

  Advertised path-id 1
  Path type: internal, path is valid, is best path
             Imported from 192.168.3.3:3:2001:DB8:1::1/64 (VRF vni100)
  AS-Path: 150 , path sourced external to AS
```

```
    ::ffff:192.168.1.0 (metric 81) from 192.168.101.101 (192.168.101.101)
      Origin incomplete, MED 0, localpref 100, weight 0
      Received label 100
      Extcommunity:
          RT:100:100
          ENCAP:8
          Router MAC:5254.004e.a437
      Originator: 192.168.1.0 Cluster list: 192.168.101.101

  Path-id 1 advertised to peers:
    30::2
```

Verification Example of BGP IPv6 Unicast configuration per VRF

```
switch(config-vrf)# show bgp vrf vni100 ipv6 unicast 2001:DB8:1::1/64
BGP routing table information for VRF vni100, address family IPv6 Unicast
BGP routing table entry for 2001:DB8::1/128, version 6
Paths: (1 available, best #1)
Flags: (0x08041e) on xmit-list, is in u6rib, is best u6rib route, is in HW
  vpn: version 7, (0x100002) on xmit-list

  Advertised path-id 1, VPN AF advertised path-id 1
  Path type: internal, path is valid, is best path
             Imported from 192.168.1.0:3:[5]:[0]:[0]:[128]:[2001:DB8:1::1]:[0::]/416
  AS-Path: 150 , path sourced external to AS
    ::ffff:192.168.1.0 (metric 81) from 192.168.101.101 (192.168.101.101)
      Origin incomplete, MED 0, localpref 100, weight 0
      Received label 100
      Extcommunity:
          RT:100:100
          ENCAP:8
          Router MAC:5254.004e.a437
      Originator: 192.168.1.0 Cluster list: 192.168.101.101

  VRF advertise information:
  Path-id 1 not advertised to any peer

  VPN AF advertise information:
  Path-id 1 not advertised to any peer
```

Verification Example of IPv4 Route configuration

```
switch(config-if)# show ip route
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>

0.0.0.0/0, ubest/mbest: 1/0
    *via vrf vni100, Null0, [20/0], 1d04h, bgp-100, external, tag 100
192.168.1.0/32, ubest/mbest: 1/0
    *via 192.168.103.1, Eth1/1, [110/81], 1d04h, ospf-100, intra
192.168.2.2/32, ubest/mbest: 1/0
    *via 192.168.103.1, Eth1/1, [110/81], 1d04h, ospf-100, intra
192.168.3.3/32, ubest/mbest: 2/0, attached
    *via 192.168.3.3, Lo0, [0/0], 1d04h, local
    *via 192.168.3.3, Lo0, [0/0], 1d04h, direct
192.168.9.9/32, ubest/mbest: 1/0, attached
    *via 192.168.9.9%vni100, Lo9, [20/0], 1d03h, bgp-100, external, tag 100
192.168.10.0/24, ubest/mbest: 1/0
    *via 192.168.1.0, [200/0], 1d04h, bgp-100, internal, tag 100 (evpn) segid: 100 tunnelid:
 0x1010101 encap: VXLAN
192.168.11.11/32, ubest/mbest: 1/0
```

```
   *via 192.168.1.0, [200/0], 1d04h, bgp-100, internal, tag 150 (evpn) segid: 100 tunnelid:
 0x1010101 encap: VXLAN
192.168.20.0/24, ubest/mbest: 1/0
   *via 192.168.2.2, [200/0], 1d04h, bgp-100, internal, tag 100 (evpn) segid: 200 tunnelid:
 0x2020202 encap: VXLAN
192.168.22.22/32, ubest/mbest: 1/0
   *via 192.168.2.2, [200/0], 1d04h, bgp-100, internal, tag 250 (evpn) segid: 200 tunnelid:
 0x2020202 encap: VXLAN
192.168.30.0/24, ubest/mbest: 1/0, attached
    *via 192.168.30.1, Eth1/2, [0/0], 1d04h, direct
192.168.30.1/32, ubest/mbest: 1/0, attached
    *via 192.168.30.1, Eth1/2, [0/0], 1d04h, local
192.168.33.0/32, ubest/mbest: 1/0
    *via 192.168.30.2, [20/0], 1d04h, bgp-100, external, tag 300
192.168.100.0/24, ubest/mbest: 1/0, attached
    *via 192.168.100.3%vni100, Vlan100, [20/0], 1d04h, bgp-100, external, tag 100
192.168.101.0/24, ubest/mbest: 1/0
    *via 192.168.103.1, Eth1/1, [110/80], 1d04h, ospf-100, intra
192.168.101.101/32, ubest/mbest: 1/0
    *via 192.168.103.1, Eth1/1, [110/41], 1d04h, ospf-100, intra
192.168.102.0/24, ubest/mbest: 1/0
    *via 192.168.103.1, Eth1/1, [110/80], 1d04h, ospf-100, intra
192.168.103.0/24, ubest/mbest: 1/0, attached
    *via 192.168.103.2, Eth1/1, [0/0], 1d04h, direct
192.168.103.2/32, ubest/mbest: 1/0, attached
```

Verification Example of IPv6 Route configuration

```
switch(config-if)# show ipv6 route
IPv6 Routing 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>

::/0, ubest/mbest: 1/0
    *via vrf vni100, Null0, [20/0], 1d04h, bgp-100, external, tag 100
2001:DB8:1::/64, ubest/mbest: 1/0, attached
    *via 2001:DB8:1::1, Eth1/1, [0/0], 1d04h, direct
2001:DB8:2:2::/128, ubest/mbest: 1/0
    *via 2001:DB8:103:1::1, Eth1/1, [110/81], 1d04h, ospf-100, intra
2001:DB8:3:3::/128, ubest/mbest: 2/0, attached
    *via 2001:DB8:3:3::3, Lo0, [0/0], 1d04h, local
    *via 2001:DB8:3:3::3, Lo0, [0/0], 1d04h, direct
2001:DB8:9:9::/128, ubest/mbest: 1/0, attached
    *via 2001:DB8:9:9::9%vni100, Lo9, [20/0], 1d03h, bgp-100, external, tag 100
2001:DB8:10::/64, ubest/mbest: 1/0
   *via 2001:DB8:1::, [200/0], 1d04h, bgp-100, internal, tag 100 (evpn) segid: 100 tunnelid:
 0x1010101 encap: VXLAN
2001:DB8:11:11::/128, ubest/mbest: 1/0
   *via 2001:DB8:1::, [200/0], 1d04h, bgp-100, internal, tag 150 (evpn) segid: 100 tunnelid:
 0x1010101 encap: VXLAN
2001:DB8:20::/64, ubest/mbest: 1/0
    *via 2001:DB8:2:2::2, [200/0], 1d04h, bgp-100, internal, tag 100 (evpn) segid: 200
tunnelid: 0x2020202 encap: VXLAN
2001:DB8:22:22::/128, ubest/mbest: 1/0
    *via 2001:DB8:2:2::2, [200/0], 1d04h, bgp-100, internal, tag 250 (evpn) segid: 200
tunnelid: 0x2020202 encap: VXLAN
2001:DB8:30::/64, ubest/mbest: 1/0, attached
    *via 2001:DB8:30::1, Eth1/2, [0/0], 1d04h, direct
2001:DB8:30::1/128, ubest/mbest: 1/0
```

Additional references

Additional references are resources or sections that provide further information related to a specific topic, such as implementing virtualization.

Related documents for VRFs

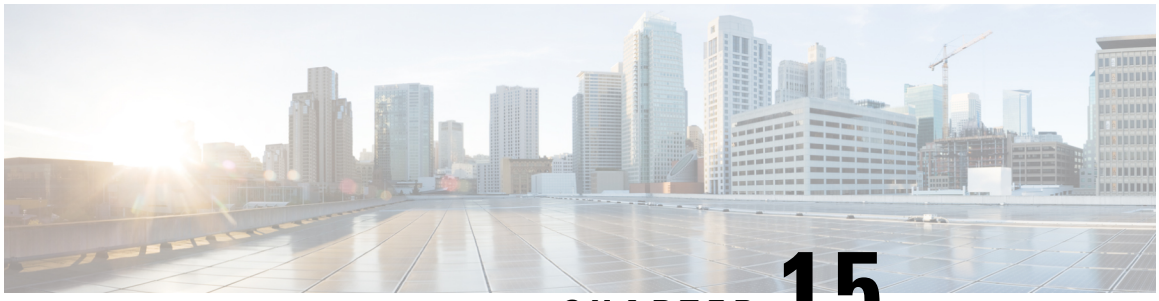
This section provides related documents for VRFs.

Related Topic	Document Title
VRFs	*Cisco Nexus 9000 Series NX-OS Fundamentals Configuration Guide* *Cisco Nexus 9000 Series NX-OS System Management Configuration Guide*

Standards

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

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



CHAPTER 15

Manage the Unicast RIB and FIB

- About the unicast RIB and FIB, on page 509
- Guidelines and limitations for the unicast RIB, on page 510
- Manage the unicast RIB and FIB, on page 511
- Verify the unicast RIB and FIB configuration, on page 520
- Additional references, on page 521

About the unicast RIB and FIB

The unicast Routing Information Base (RIB) and Forwarding Information Base (FIB) are integral parts of the Cisco NX-OS forwarding architecture. The RIB maintains the routing table and determines the best next hop for routes, while the FIB is populated with forwarding information for efficient packet delivery.

- The unicast RIB exists on the active supervisor and maintains the routing table with directly connected, static, and dynamically learned routes.
- The unicast RIB collects adjacency information from sources such as the Address Resolution Protocol (ARP).
- The unicast RIB determines the best next hop for each route and populates the unicast FIBs on the modules using the unicast FIB distribution module (FDM).

Unicast RIB and FIB in Cisco NX-OS forwarding architecture

The unicast RIB and FIB are essential for the operation of the Cisco NX-OS forwarding architecture, enabling efficient routing and forwarding of unicast traffic.

- Directly connected routes, static routes, and routes learned from dynamic unicast routing protocols are maintained in the unicast RIB.
- Adjacency information is collected from ARP and other sources.
- The unicast FIB is populated by the RIB and used for forwarding decisions on the modules.

1. Each dynamic routing protocol updates the unicast RIB when a route times out.
2. The unicast RIB deletes the expired route and recalculates the best next hop if an alternate path is available.

Unicast RIB and FIB operation

For example, when a dynamic routing protocol detects that a route has timed out, it updates the unicast RIB. The RIB then removes the route and, if another path exists, recalculates the best next hop and updates the FIB accordingly.

Layer 3 consistency checker

The Layer 3 consistency checker is a feature in Cisco NX-OS that detects inconsistencies between the unicast IPv4 RIB on the supervisor module and the FIB on each interface module.

- Detects missing prefix, extra prefix, wrong next-hop address, and incorrect Layer 2 rewrite string in the ARP or neighbor discovery (ND) cache.
- Compares FIB entries to the latest adjacency information from the Adjacency Manager (AM) and logs any inconsistencies.
- Compares unicast RIB prefixes to the module FIB and logs any inconsistencies.

Layer 3 consistency checker operation and behavior

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 to detect and log such inconsistencies.

- 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.

When more routes are learned exceeding the hardware limit, the **show consistency-checker forwarding ipv4** command is run, consistency may still show as pass. The same is true when it is transitioning from an inconsistent state to a consistent state. It may show as a failure. Until and unless the **test forwarding ipv4 inconsistency route** command is run again, it doesn't leave this state. This is an expected behavior.

Guidelines and limitations for the unicast RIB

The following guidelines and limitations apply to the URIB or U6RIB.

In a virtual domain context (VDC), when modifying memory resource limits for the IPv4 or IPv6 unicast route, the modified limits do not take effect immediately. You must issue the **copy running-config startup-config** command followed by the **reload** command to activate the modified limits. For example, if

you issue either of the following commands, you will need to issue **copy running-config startup-config**, then reload the switch an extra time to activate the new setting:

- **limit-resource u4route-mem**
- **limit-resource u6route-mem**



Note If "feature pim" is configured for limit-resource, ensure that the value of **limit-resource u4route-mem** plus **limit-resource u6route-mem** is <= 1024 MB (1GB).

Beginning with Cisco NX-OS Release 10.3(1)F, Unicast consistency checker is supported on Cisco Nexus 9808 platform switches.

- Beginning with Cisco NX-OS Release 10.4(1)F, Unicast consistency checker is supported on Cisco Nexus X98900CD-A and X9836DM-A line cards with Cisco Nexus 9808 switches.

Beginning with Cisco NX-OS Release 10.4(1)F, Unicast consistency checker is supported on Cisco Nexus 9804 platform switches, and Cisco Nexus X98900CD-A and X9836DM-A line cards.

Beginning with Cisco NX-OS Release 10.5(1)F, the Layer 3 ECMP Dynamic Load Balancing (DLB) feature provides support to efficiently load balance traffic, depending on the current state of utilization of the outgoing links.

Manage the unicast RIB and FIB

Manage the Unicast RIB and FIB refers to the process of handling the Routing Information Base (RIB) and Forwarding Information Base (FIB) in a network device.

- The RIB stores routing information learned from different protocols.
- The FIB contains the actual forwarding table used by the device.
- Management involves updating, maintaining, and synchronizing both tables for correct packet forwarding.

Differences in Command Syntax

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

Display the module FIB information

To display the FIB information on a module, use the following commands in any mode:

Procedure

Step 1 `show forwarding { ipv4 | ipv6 } adjacency module slot`

Example:

```
switch# show forwarding ipv6 adjacency module 2
```

Displays the adjacency information for IPv4 or IPv6.

Step 2 **show forwarding** { **ipv4** | **ipv6** } **route module** *slot*

Example:

```
switch# show forwarding ipv6 route module 2
```

Displays the route table for IPv4 or IPv6.

Configure 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.

Note Load sharing uses the same 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:

Procedure

Step 1 **ip load-sharing address** { **destination port destination** | **source-destination** [**port source-destination]** | **source** } [**exclude-l3-proto**] **hardware lb-keyshift** *value* **lb-2nd-heir-keyshift** *value* [**universal-id** *seed*] [**rotate** *rotate*] [**concatenation**]

Example:

```
ip load-sharing address source-destination port source-destination hardware lb-keyshift 1
lb-2nd-hier-keyshift 10
```

Configures the unicast FIB load-sharing algorithm for data traffic.

Note On Cisco Nexus 9808 /9804 switches, only **address source-destination port source-destination** option is supported during **ip load-sharing address** configuration.

Beginning with Cisco NX-OS Release 10.3(3)F, the **hardware** option is added to support the following parameters in the **IHB_ECMP_LB_KEY_CFG** tables only on Cisco Nexus 9600-R/RX line cards:

- **lb-keyshift** : Sets the **ECMP_LB_KEY_SHIFT** value for load balancing. The range is 1-10.
- **lb-2nd-hier-keyshift** : Sets the **ECMP_2ND_HIER_LB_KEY_SHIFT** value for load balancing. The range is 1-10.

Beginning with Cisco NX-OS Release 10.5(1)F, the **exclude-l3-proto** option has been introduced. This option allows the exclusion of the IP protocol from ECMP hashing during next-hop selection on Cisco Nexus 9300-GX2 Series switches.

The following options are available for all IP load sharing configurations:

- The **universal-id** option sets the random seed for the hash algorithm and shifts the flow from one link to another.

  You do not need to configure the universal ID. Cisco NX-OS chooses the universal ID if you do not configure it. The *universal-id* range is from 1 to 4294967295.

- The **rotate** option causes the hash algorithm to rotate the link picking selection so that it does not continually choose the same link across all nodes in the network. It does so by influencing the bit pattern for the hash algorithm. This option shifts the flow from one link to another and load balances the already load-balanced (polarized) traffic from the first ECMP level across multiple links.

  If you specify a *rotate* value, the 64-bit stream is interpreted starting from that bit position in a cyclic rotation. The *rotate* range is from 1 to 63, and the default is 32.

  Note With multi-tier Layer 3 topology, polarization is possible. To avoid polarization, use a different rotate bit at each tier of the topology.

  Note To configure a rotation value for port channels, use the **port-channel load-balance src-dst ip-l4port rotate** *rotate* command. For more information on this command, see the *Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide* .

- The **concatenation** option ties together the hash tag values for ECMP and the hash tag values for port channels in order to use a stronger 64-bit hash. If you do not use this option, you can control ECMP load-balancing and port-channel load-balancing independently. The default is disabled.

Step 2 (Optional) **show ip load-sharing**

Example:

```
switch(config)# show ip load-sharing
address source-destination
```

Displays the unicast FIB load-sharing algorithm for data traffic.

Step 3 (Optional) **show routing hash** *source-addr dest-addr* [*source-port dest-port*] [**vrf** *vrf-name*]

Example:

```
switch(config)# show routing hash 192.0.2.1
10.0.0.1
```

Displays the route that the unicast RIB and unicast 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.

This example shows how to display the route selected for a source/destination pair:

```
switch#
show routing hash 10.0.0.5 192.0.0.2
Load-share parameters used for software forwarding:
load-share mode: address source-destination port source-destination
Universal-id seed: 0xe05e2e85
Hash for VRF "default"
Hashing to path *172.0.0.2 (hash: 0x0e), for route:
```

This example shows the output of **show ip load-sharing** command:

```
switch(config)# show ip load-sharing
IPv4/IPv6 ECMP load sharing:
Universal-id (Random Seed): 1913447906
Load-share mode : address source

Exclude L3 proto from ECMP hashing : Enabled

Rotate: 32
switch(config)#
```

Dynamic load balancing

Dynamic load balancing is a mechanism that distributes network traffic across multiple paths to optimize resource utilization and performance.

- Enables efficient use of available network bandwidth.
- Improves redundancy and fault tolerance by balancing traffic loads.
- Adapts to changing network conditions for optimal routing.

The contents of the *Dynamic Load Balancing* section have been moved from the *Managing the Unicast RIB and FIB* chapter to a new chapter, Configure Dynamic Load Balancing , in the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide itself from 10.5(3)F, specifically to the Dynamic Load Balancing on CloudScale switches section.

Display the routing and adjacency information

To display routing and adjacency information, use the following commands in any mode:

Command	Purpose
show { **ip** | **ipv6** } **route** [*route-type* | **interface** *interface-type number* | **next-hop**] `switch# show ip route`	Displays the unicast route table. The *route-type* argument can be a single route prefix or a direct, static, or dynamic route protocol. Use the **?** command to see the supported interfaces.
show { **ip** | **ipv6** } **adjacency** [*prefix* | *interface-type 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 IPv4 or IPv6 prefix address. • *interface-type number* —Use the **?** command to see the supported interfaces. • *vrf-id* —Any case-sensitive, alphanumeric string up to 64 characters.
show { **ip** | **ipv6** } **routing** [*route-type* | **interface** *interface-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 or a direct, static, or dynamic route protocol. Use the **?** command to see the supported interfaces.

Procedure

Enter the appropriate **show** command to display routing or adjacency information.

Example:

```
switch# show ip route
```

This example shows how to display the unicast route table:

```
switch# show ip route
IP Route Table for Context "default"
'*' denotes best ucast next-hop '**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
0.0.0.0/0, 1 ucast next-hops, 0 mcast next-hops
*via 10.1.1.1, mgmt0, [1/0], 5d21h, static
0.0.0.0/32, 1 ucast next-hops, 0 mcast next-hops
*via Null0, [220/0], 1w6d, local, discard
10.1.0.0/22, 1 ucast next-hops, 0 mcast next-hops, attached
*via 10.1.1.55, mgmt0, [0/0], 5d21h, direct
10.1.0.0/32, 1 ucast next-hops, 0 mcast next-hops, attached
*via 10.1.0.0, Null0, [0/0], 5d21h, local
10.1.1.1/32, 1 ucast next-hops, 0 mcast next-hops, attached
*via 10.1.1.1, mgmt0, [2/0], 5d16h, am
10.1.1.55/32, 1 ucast next-hops, 0 mcast next-hops, attached
*via 10.1.1.55, mgmt0, [0/0], 5d21h, local
```

```
10.1.1.253/32, 1 ucast next-hops, 0 mcast next-hops, attached
*via 10.1.1.253, mgmt0, [2/0], 5d20h, am
10.1.3.255/32, 1 ucast next-hops, 0 mcast next-hops, attached
*via 10.1.3.255, mgmt0, [0/0], 5d21h, local
255.255.255.255/32, 1 ucast next-hops, 0 mcast next-hops
*via Eth Inband Port, [0/0], 1w6d, local
```

This example shows how to display the adjacency information:

```
switch#
show ip adjacency
IP Adjacency Table for context 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
```

Trigger 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:

Procedure

Step 1 **test forwarding** [**ipv4** | **ipv6**] [**unicast**] **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 64 characters. The *slot* range is from 1 to 26.

Step 2 **test forwarding** [**ipv4** | **ipv6**] [**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 26.

Step 3 **show forwarding** [**ipv4** | **ipv6**] [**unicast**] **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 64 characters. The *slot* range is from 1 to 26.

Step 4 **show consistency-checker forwarding unicast**

Example:

```
switch(config)# show consistency-checker forwarding unicast
```

Displays the results of a Layer 3 consistency check for unicast routes.

Clear the FIB forwarding information

You can clear one or more entries in the FIB. Clearing a FIB entry does not affect the unicast RIB.



Caution The **clear forwarding** command disrupts forwarding on the device.

To clear an entry in the FIB, including a Layer 3 inconsistency, use the following command in any configuration mode:

Procedure

clear forwarding { **ipv4** | **ipv6** } **route** { ***** | *prefix* } [**vrf** *vrf-name*] **module** { *slot* | **all** }

Example:

```
switch# clear forwarding ipv4 route * module 1
```

Clears one or more entries from the FIB. The route options are as follows:

- ***** —All routes.
- *prefix* —All routes.

The *vrf-name* can be any case-sensitive, alphanumeric string up to 64 characters. The *slot* range is from 1 to 26.

Configure maximum routes for the unicast RIB

You can configure the maximum number of routes allowed in the routing table.

Procedure

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 management2
switch(config-vrf)#
```

Creates a VRF and enters VRF configuration mode.

Step 3 **address-family {ipv4 | ipv6} unicast**

Example:

```
switch(config-vrf)# address-family ipv4 unicast
switch(config-vrf-af-ipv4)
```

Enters the address-family configuration mode.

Step 4 **maximum routes** *max-routes* [*threshold* [**reinstall** *threshold*] | **warning -only**]

Example:

```
switch(config-vrf-af-ipv4)# maximum routes 300000
```

Configures the maximum number of routes allowed in the routing table. The range is from 1 to 4294967295.

You can optionally specify the following:

- *threshold* —Percentage of maximum routes that triggers a warning message. The range is from 1 to 100.
- **warning-only** —Logs a warning message when the maximum number of routes is exceeded.
- **reinstall** *threshold* —Reinstalls routes that previously exceeded the maximum route limit and were rejected and specifies the threshold value at which to reinstall them. The threshold range is from 1 to 100.

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

Example:

```
switch(config-vrf-af-ipv4)# copy running-config
startup-config
```

Saves this configuration change.

Estimate memory requirements for routes

To estimate the memory requirements for routes, use the following command in any mode:

Command	Purpose
show routing { **ipv6** } **memory estimate routes** *num-routes* **next-hops** *num-nexthops* **Example:** `switch# show routing memory estimate routes 5000 next-hops 2`	Displays the memory requirements for routes. The *num-routes* range is from 1000 to 1000000. The *num-nexthops* range is from 1 to 16.

Procedure

Enter the **show routing** { **ipv6** } **memory estimate routes** *num-routes* **next-hops** *num-nexthops* command.

Example:

```
switch# show routing memory estimate routes 5000 next-hops 2
```

Displays the memory requirements for routes. The *num-routes* range is from 1000 to 1000000. The *num-nexthops* range is from 1 to 16.

Clear 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 configuration mode:

Procedure

Step 1 **clear** { **ip** | **ip4** | **ipv6** } **route** { ***** | { *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 IP or IPv6 route.
- *prefix* / *length* —Any IP or IPv6 prefix.
- *next-hop* —The next-hop address.
- *interface* —The interface to reach the next-hop address.

The *vrf-name* can be a case-sensitive, alphanumeric string up to 64 characters.

Step 2 **clear routing** [**multicast** | **unicast**] [**ip** | **ip4** | **ipv6**] { ***** | { *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 IP or IPv6 route.
- *prefix* / *length* —Any IP or IPv6 prefix.
- *next-hop* —The next-hop address.
- *interface* —The interface to reach the next-hop address.

The *vrf-name* can be a case-sensitive, alphanumeric string up to 64 characters.

Verify the unicast RIB and FIB configuration

To display the unicast RIB and FIB configuration information, perform one 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 {**ip** | **ipv4** | **ipv6**} **route**	Displays routes in the FIB.
show {**ip** | **ipv6**} **adjacency**	Displays the adjacency table.
show {**ip** | **ipv6**} **route**	Displays the IPv4 or IPv6 routes from the unicast RIB.
show routing	Displays routes from the unicast RIB.
show system internal access-list dest-miss stats	Displays statistics for packets dropped due to missing the FIB routes for the destinations, also called as DEST MISS. The output displays increment in the DEST MISS counters. **Note** Beginning with Cisco NX-OS Release 10.1(1), this feature is supported on Cisco Nexus 9300-FX3 platform switches.

Additional references

For additional information related to managing unicast RIB and FIB, see the following table.

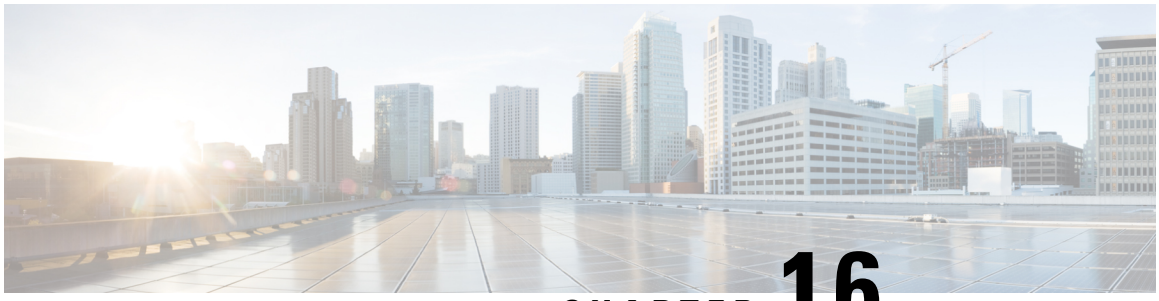
Related Topic	Document Title
Configuring EEM	Cisco Nexus 9000 Series NX-OS System Management Configuration Guide

Related documents

This section provides a reference to related documents for configuring EEM.

Table 34: Related documents

Related Topic	Document Title
Configuring the Embedded Event Manager	Cisco Nexus 9000 Series NX-OS System Management Configuration Guide



CHAPTER 16

Configuring Route Policy Manager

This chapter contains the following sections:

- About route policy manager, on page 523
- Route map support matrix for routing protocols, on page 534
- Guidelines and limitations for route policy manager, on page 537
- Default settings for route policy manager parameters, on page 538
- Configure route policy manager, on page 539
- Global commands to block the deletion of route-map, on page 552
- Verify the route policy manager configuration, on page 553
- Configuration examples for route policy manager, on page 553
- Related topics, on page 553

About route policy manager

Route Policy Manager is a feature that supports route maps and IP prefix lists for route redistribution and filtering.

- Supports route maps and IP prefix lists.
- Enables route redistribution and filtering between routing domains.
- Prefix lists contain IPv4 or IPv6 network prefixes and associated prefix length values.

Route Policy Manager enables the use of route maps and prefix lists for advanced routing control.

- Route maps can apply to both routes and IP packets.
- Prefix lists can be used in BGP templates, route filtering, or redistribution of routes exchanged between routing domains.

Prefix lists

Prefix lists are a method to filter network routes or packets by matching their prefixes against a defined list of permitted or denied prefixes.

- Permit or deny an address or range of addresses based on prefix matching.
- Multiple entries can be configured, each with an associated sequence number.

- Evaluation starts with the lowest sequence number, and processing stops after the first match.

How prefix lists work

Filtering by a prefix list involves matching the prefixes of routes or packets with the prefixes listed in the prefix list. If a given prefix does not match any entries, an implicit deny is assumed.

Note An empty prefix list permits all routes.

MAC lists

A MAC list is a collection of MAC addresses and optional MAC masks used to permit or deny network traffic based on MAC address matching.

- Consists of MAC addresses and optional wild-card MAC masks.
- Used to permit or deny MAC addresses or ranges of addresses.
- Filtering involves matching packet MAC addresses with entries in the MAC list; unmatched addresses are implicitly denied.

How MAC lists work

MAC lists are evaluated in sequence, and each entry has an associated sequence number. Cisco NX-OS processes the first successful match for a given MAC address and applies the permit or deny action. Once a match occurs, the rest of the MAC list is not evaluated for that address.

MAC list usage

For example, you can configure multiple entries in a MAC list to permit or deny specific MAC addresses. Each entry is assigned a sequence number, and the system evaluates entries starting with the lowest sequence number. The first matching entry determines whether the MAC address is permitted or denied.

Route maps

Route maps are a category of configuration tools that allow you to control route redistribution by specifying match and set criteria for routes or packets.

- Each route map entry includes a sequence number to determine processing order.
- Entries specify permission (permit or deny), match criteria, and set changes.
- Route maps can process entries in a linear or user-defined order using the **continue** statement.

Route map structure and processing

Route maps are composed of one or more entries, each identified by a sequence number under a unique route map name. Each entry defines how routes or packets are matched and what actions are taken.

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). Route map can be confgured to process in a different order using the continue statement, which determines the route map entry that needs to be processed next.

Default action for sequences in a route map

The default action for any sequence in a route map is **permit.**

- If you configure a new sequence in a route map without explicitly specifying either **permit** or **deny.**, the default action is **permit.**
- If you edit a configured sequence in a route map and do not specify an action, the **permit** action is applied, even if the sequence was originally configured with **deny.**
- Always set the correct action when configuring or editing a sequence of a route map; otherwise, the default action, **permit**, is applied.

Default sequence number for a route map

The default sequence number for a route-map with no specified sequence value is 10.

- If you create a new route-map without specifying a sequence number, the default sequence number is 10.
- If a route-map already exists with sequence number 10 and you configure the same route-map again without specifying a sequence number, any modifications will be applied to sequence number 10 of that route-map.
- If a route-map already has sequence numbers assigned (20, 30, 40, etc.) and you configure it again without specifying a sequence number, a new entry with sequence number 10 will be created for that route-map.

Match criteria

Match criteria are the set of parameters used to determine whether a route or IP packet meets specific conditions in a route map.

- Some criteria, such as BGP community lists, are applicable only to a specific routing protocol.
- Other criteria, such as the IP source or destination address, can be used for any route or IP packet.
- Match criteria are evaluated by comparing the route or packet to each configured match statement in the route map.

Types of match criteria and processing behavior

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 rendezvous point, groups, or sources.
- Other parameters—Match based on IP next-hop address or packet length.

For match processing:

1. If multiple match statements of the same type exist within the same route-map sequence, they are processed as an OR operation. This processing applies whether the match statements are on the same line or not.
2. If multiple match statements of a different type exist within the same route-map sequence, they are processed as an AND operation.

Set changes

Set changes are modifications applied to a route or packet after it matches an entry in a route map, based on configured set statements.

- Change BGP parameters such as AS-path, tag, community, extended community, dampening, local preference, origin, or weight attributes.
- Change metrics, including the route-metric or the route-type.
- Change other parameters, such as the forwarding address or the IP next-hop address.

Set changes are used in route maps to modify route or packet attributes after a match occurs.

Access lists

- IP access lists can match packets to fields such as source or destination IPv4 or IPv6 address.
- They can match on protocol, precedence, and ToS.
- Access lists can be used in a route map for policy-based routing only.

AS numbers for BGP

AS numbers for BGP are identifiers that allow BGP to match peers and establish sessions based on configured Autonomous System numbers.

- You can configure a list or range 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.

BGP uses AS numbers to determine whether to establish a session with a peer. You can configure these as a list, a range, or use an AS-path list with a regular expression.

AS-path lists for BGP

An AS-path list is a configuration tool that allows filtering of BGP route updates based on the AS-path attribute.

- Filters inbound or outbound BGP route updates using AS-path attributes.
- Processes routes according to permit or deny conditions configured in the AS-path list.
- Supports multiple AS-path entries under the same list name; the router processes the first matching entry.

AS-path list configuration and processing in 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. Multiple AS-path entries can be configured 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

Community lists for BGP are mechanisms that allow filtering and matching of BGP route updates based on the community attribute using route maps.

- Community lists can match the community attribute in BGP routes and set the community attribute using route maps.
- A community list contains one or more community attributes; all must match for a route to be considered a match within a single entry.
- Multiple community attributes can be configured as individual entries with the same community list name, and the router processes the first matching entry according to its permit or deny action.

Community list formats and usage in BGP

Community attributes in a community list can be configured in several formats to match BGP routes as needed.

- 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 for BGP are a category of access lists that support 4-byte AS numbers and allow configuration of community attributes in specific formats.

- Support 4-byte AS numbers for BGP community attributes.
- Allow configuration in *aa4:nn* format, where the first four bytes represent the AS number and the last two bytes represent a user-defined network number.
- Permit use of regular expressions for matching community attributes.

Properties and behavior of extended community lists

Extended community lists in Cisco NX-OS provide similar functionality to regular community lists for four-byte AS numbers and can be configured with specific properties.

- Transitive: BGP propagates the community attributes across autonomous systems.
- Nontransitive: BGP removes community attributes before propagating the route to another autonomous system.

Configure NX-OS BGP large communities

About NX-OS BGP large communities

NX-OS BGP large communities are standardized 12-byte values (per IETF RFC 8092) that provide enhanced flexibility for classifying BGP routes, especially when using 4-byte ASNs.

- Allow classification of routes from different data centers and ASNs using large communities.
- Remove the 4-byte restriction of standard BGP communities by supporting 12-byte large communities.
- Enable more flexible configuration of networks and routing policies in NX-OS BGP.

Beginning with Cisco NX-OS Release 10.6(2)F, BGP large community is supported on EVPN address-family. In earlier releases, it is supported on IPv4 and IPv6 unicast address family.

Configure a large community list (expanded)



Note When you define an community-list regular expression in an NDFC or DCNM configuration profile, enclose the expression in double quotation marks. NX-OS stores the expression in quoted form. An equivalent unquoted expression in the profile can differ from the running configuration after a reload, causing profile application to fail because of conflicting commands.

The following are the steps to configure large community list in expanded form:

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **ip large-community-list** *expanded*

Example:

```
switch(config)# ip large-community-list
                        expanded
```

This option adds an expanded large community list entry.

Step 3 **ip large-community-list expanded** *list-name*

Example:

```
switch(config)# ip large-community-list expanded
                      list-name
```

This option provides the name of the expanded large community list. The *list-name* can be any case-sensitive, alphanumeric string up to 63 characters.

Step 4 **ip large-community-list expanded abcd** *seq*

Example:

```
switch(config)# ip large-community-list expanded abcd
                      seq
```

This option provides the sequence number of the entry.

Step 5 **ip large-community-list expanded abcd seq 10** {*deny* | *permit*}

Example:

```
switch(config)# ip large-community-list expanded abcd seq 10
                      {deny | permit}
```

The first option specifies the large community to reject.

The second option specifies the large community to accept.

Step 6 **ip large-community-list expanded abcd seq 10 permit** *XX:YY:ZZ*

Example:

```
switch(config)# ip large-community-list expanded abcd seq 10 permit
                      XX:YY:ZZ
```

This option provides the regular expression which uses a XX:YY:ZZ format. XX can have a range of <0-4294967294> and is a four octet global administrator field which represents ASN. Whereas, YY and ZZ are four octet local data fields, which are defined by an owner of the ASN.

The ":" is a separator between global and local data fields.

The following example shows how to create a large community list in expanded form:

```
switch(config)# ip large-community-list expanded abcd seq 10 permit ”^100:200:300$"
               switch(config)# sh run rpm
               <<SNIP>>
               ip large-community-list expanded abcd seq 10 permit ”^100:200:300$"
```

Configure a large community list (standard)

The following are the steps to configure large community list in standard form:

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **ip large-community-list** *standard*

Example:

```
switch(config)# ip large-community-list
                        standard
```

This option adds a standard large community list entry.

Step 3 **ip large-community-list standard** *list-name*

Example:

```
switch(config)# ip large-community-list standard
                        list-name
```

This option provides the name of the standard large community list. The *list-name* can be any case-sensitive, alphanumeric string up to 63 characters.

Step 4 **ip large-community-list standard efgh** *seq*

Example:

```
switch(config)# ip large-community-list standard efgh
                        seq
```

This option provides the sequence number of the entry.

Step 5 **ip large-community-list standard efgh seq 15** {*deny* | *permit*}

Example:

```
switch(config)# ip large-community-list standard efgh seq 15
                        {deny | permit}
```

The first option specifies the large community to reject.

The second option specifies the large community to accept.

Step 6 **ip large-community-list standard efgh seq 15 deny** *XX:YY:ZZ*

Example:

```
switch(config)# ip large-community-list standard efgh seq 15 deny
                        XX:YY:ZZ
```

This option provides the regular expression which uses a XX:YY:ZZ format. XX can have a range of <0-4294967294> and is a four octet global administrator field which represents ASN. Whereas, YY and ZZ are four octet local data fields, which are defined by an owner of the ASN.

The ":" is a separator between global and local data fields.

The following example shows how to create a large community list in standard form:

```
switch(config-route-map)# ip large-community-list standard efgh seq 15 deny 1000300:123:456

                    switch(config)# sh run rpm
```

```
<<SNIP>>
ip large-community-list standard efgh seq 15 deny 1000300:123:456
```

Configure a route-map match for large community

The following are the steps to configure route-map match for large community:

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **route-map** *mape-name* [**permit | deny**] *sequence-number*

Example:

```
switch# route-map test permit 10
```

Enters the route-map configuration mode and defines the conditions for routes from one routing protocol into another.

Step 3 **match** *large-community*

Example:

```
switch(config-route-map)# match
                      large-community
```

This option matches BGP large community list.

Step 4 **match large-community** *list-name*

Example:

```
switch(config-route-map)# match large-community
                      list-name
```

This option provides the name of the community list. The *list-name* can be any case-sensitive, alphanumeric string up to 63 characters.

Step 5 **match large-community abcd** *exact-match*

Example:

```
switch(config-route-map)# match large-community abcde
                      exact-match
```

This option does the exact matching of the communities.

The following example shows how to create a large community list in expanded form:

```
switch(config-route-map)# sh run rpm
                <<SNIP>>
                route-map test permit 10
```

```
match large-community abcd efgh
```

Configure a route map set for large community

The following are the steps to configure route-map set for large community:

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **route-map** *mape-name* [**permit | deny**] *sequence-number*

Example:

```
switch# route-map test permit 10
```

Enters the route-map configuration mode and defines the conditions for routes from one routing protocol into another.

Step 3 **set** *large-community-list*

Example:

```
switch(config-route-map)# set
                        large-community-list
```

This option sets BGP large community attribute.

Step 4 **set large-community-list** *list-name*

Example:

```
switch(config-route-map)# set large-community-list
                        list-name
```

This option sets the name of the large community list. The *list-name* can be any case-sensitive, alphanumeric string up to 63 characters.

Step 5 **set large-community-list list-name** *delete*

Example:

```
switch(config-route-map)# set large-community-list list-name
                        delete
```

This option deletes the matching large communities.

Example:

```
switch(config-route-map)# sh run rpm
                        route-map test permit 10
                        set large-community-list list-name delete
```

Step 6 **set large-community** {*none* | *XX:YY:ZZ* [*additive*] | *additive*}

Example:

```
switch(config-route-map)# set large-community
                          {none | XX:YY:ZZ [additive] | additive}

switch(config-route-map)# set large-community
                          1000:1235:7629 200:30048:234 additive
```

This command sets the large-community attribute for a BGP route update.

- The 'XX:YY:ZZ' option represents the large-community attribute in XX:YY:ZZ format and sets that value alone for a BGP route update. A maximum of 32 large-community attributes can be added in one set command.
- The 'additive' option represents an addition to the existing large-community attribute, and is used along with the XX:YY:ZZ option. When used in this manner, it adds the XX:YY:ZZ attribute to the existing large-community attribute.
- The 'none' option represents that no large-community attribute will be set.

Example:

```
switch(config-route-map)# sh run rpm
                          route-map test permit 10
                          set large-community additive

switch(config-route-map)# sh run rpm
                          route-map test permit 10
                          set large-community 1000300:123:456

switch(config-route-map)# sh run rpm
                          route-map test permit 10
                          set large-community none
```

Route redistribution and route maps

Route redistribution with route maps is a process that controls which routes are redistributed between routing domains and how their attributes are modified.

- Route maps match on route attributes to selectively redistribute routes that meet specific criteria.
- Route maps can modify route attributes during redistribution using set actions.
- Routes are evaluated against each route map entry or sequence until a match is found or all entries are processed.

How route maps control route redistribution

Route maps provide granular control over which routes are redistributed and how their attributes are set during the redistribution process.

- Routes are matched against each route map entry or sequence.
- If multiple match statements exist under a route-map sequence, the route must satisfy all match criteria.
- If a route matches the criteria, the set actions are executed.
- If a route does not match, it is compared against subsequent route map entries or sequences.
- If no match is found after all entries are processed, the route is denied (either acceptance for inbound or forwarding for outbound route maps).

Note When redistributing BGP to IGP, iBGP routes are redistributed by default. To override this behavior, insert an additional deny statement into the route map.

Route map support matrix for routing protocols

The item is a category - name that includes configurable match and set statements for routing protocols on Cisco Nexus 9000 Series switches.

- Yes—The statement is supported for the protocol.
- No—The statement is not supported for the protocol.
- If a statement does not apply for the protocol, there is an em dash (—) in the column next to the statement.
- Where clarification is required, information is added in the appropriate row/column.

Table 35: SET route map statements by protocol

SET Route Map Statement	OSPF Redistribution	EIGRP Redistribution	ISIS Redistribution	RIP Redistribution	BGP Redistribution
Forwarding-address	Yes	—	—	—	—
Standard/Extended Community	—	—	—	—	Standard community only
Site of Origin (SOO)	—	—	—	—	No
Routing Protocol Metric	Yes	Yes	Yes	Yes	Yes
Routing Protocol Metric Type	Yes	—	No	—	—
Route Tag	Yes	Yes	No	Yes	—
NSSA Only	Yes	—	—	—	—
Orgin	—	—	—	—	Yes
Level	—	—	Yes	—	—
Weight	—	—	—	—	Yes

Table 36: SET route map statements by protocol

SET Route Map Statement	BGP Neighbor	BGP Table Map	OSPF Table Map	EIGRP Table Map	ISIS Table Map	EIGRP Distribute List
Standard/Extended Community	Yes	No	—	—	—	—
Standard/Extended Community-List Deletion	Yes	No	—	—	—	—
Site of Origin (SOO)	No	—	—	—	—	—
Routing Protocol Metric	No	No	—	—	—	Yes
Routing Protocol Metric Type	Yes	No	—	—	—	—
IPv4 Next Hop	Yes	—	—	—	—	—
IPv6 Next Hop	Yes	—	—	—	—	—
IPv4 Prefix list	Yes	—	—	—	—	—
IPv6 Prefix list	Yes	—	—	—	—	—
Interface	No	—	—	—	—	—
Route Tag	—	—	—	—	—	Yes
AS PATH	Yes	No	—	—	—	—
Orgin	Yes	No	—	—	—	—
All Path Advertisement	Yes	No	—	—	—	—
Distance	—	Yes	Yes	Yes	Yes	—
Dampening	No	No	—	—	—	—
Level	No	No	—	—	—	—
Weight	Yes	Yes	No	—	—	—

Table 37: MATCH route map statements by protocol

MATCH Route Map Statement	OSPF Redistribution	EIGRP Redistribution	ISIS Redistribution	RIP Redistribution	BGP Redistribution
Community List	OSPFv2 only	Yes	yes	Yes	—

MATCH Route Map Statement	OSPF Redistribution	EIGRP Redistribution	ISIS Redistribution	RIP Redistribution	BGP Redistribution
Ext Community List	OSPFv2 only	Yes	—	Yes	—
Interface	Yes	Yes	Yes	Yes	Yes
IPv4 Next Hop	Yes	Yes	Yes	Yes	Yes
IPv6 Next Hop	Yes	Yes	Yes	No	Yes
Metric	Yes	Yes	Yes	Yes	Yes
Route Type	Yes	Yes	Yes	Yes	Yes
Tag	Yes	Yes	Yes	Yes	Yes
IPv6 Prefix List	Yes	Yes	Yes	No	Yes
IPv4 Prefix list	Yes	Yes	Yes	Yes	Yes
IP ACL	No	No	No	No	No
Source Protocol	Yes	Yes	Yes	Yes	—
AS Path	No	No	No	No	—
AS Number	No	No	No	No	—

Table 38: MATCH route map statements by protocol

MATCH Route Map Statement	BGP Neighbor	BGP Table Map	OSPF Table Map	EIGRP Table Map	ISIS Table Map	EIGRP Distribute List
Community List	Yes	Yes	—	—	—	—
Ext Community List	Yes	Yes	—	—	—	—
Interface	—	No	Yes	Yes	Yes	—
IPv4 Next Hop	Yes	Yes	Yes	Yes	Yes	Yes
IPv6 Next Hop	Yes	Yes	Yes	Yes	Yes	Yes
Metric	Yes	Yes	Yes	No	No	No
Route Type	Yes	Yes	Yes	Yes	Yes	No

MATCH Route Map Statement	BGP Neighbor	BGP Table Map	OSPF Table Map	EIGRP Table Map	ISIS Table Map	EIGRP Distribute List
Tag	—	Yes	Yes	Yes	No	Yes
IPv6 Prefix List	Yes	Yes	Yes	Yes	Yes	Yes
IPv4 Prefix list	Yes	Yes	Yes	Yes	Yes	Yes
IP ACL	No	No	No	No	No	No
AS Path	Yes	Yes	—	—	—	—
AS Number	Yes	No	—	—	—	—
IPv4 Route Source	—	—	Yes	—	—	—

Guidelines and limitations for route policy manager

Route Policy Manager enforces configuration guidelines and limitations to ensure correct routing policy behavior. Improper use of unsupported commands or ambiguous configurations can result in unintended routing outcomes. Specific behaviors apply to route-maps, prefix lists, ACLs, and policy application in Cisco NX-OS.

Configuration guidelines and limitations

- Although the command 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. It is recommended to use unique names. Do not use the same name by modifying upper-case and lowercase 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.
- When matching two irrelevant entities in the route-map entry, the permission (permit or deny) of the route-map entry decides the result for all the routes or packets. It also applies the set criteria of the route-map entry. For example, the following route-map, when associated with the BGP configuration, tries to match the ospf-area which results in permitting the irrelevant match and sets the metric to 100:

```
route-map abc permit seq 10
match ospf-area 2
set metric 100
```

- 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 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.
- If you configure standard ip community-list and ip large-community-list in multiple lines in config-profile, only the last configured line of that sequence persists. To execute these 2 commands, you need to configure all the community values and execute as a single command in config-profile.
- If policy is applied via config profile, it is not preferred to attempt unconfiguration (with short no form) of the particular CLI via normal CLI configuration mode. If any changes are required, unapply the profile first, and then modify the profile and apply again.
- For any RPM profile, if you're planning to configure and apply the config profile ensure not to configure and unconfigure (with short no form) the same profile, if you wish to use "config profile" later.
- Beginning with Cisco NX-OS Release 10.2(2)F, matching on tags for BGP NLRI (for inbound and outbound facing route-maps) is now supported. However, this is only intended for the use of the L2VPN EVPN address family in L4-7 service integration in VXLAN.
- Beginning with Cisco NX-OS Release 10.6(2)F, BGP large community is supported on EVPN address-family. In the earlier releases, it is supported on IPv4 and IPv6 unicast address family.

Default settings for route policy manager parameters

The following table lists the default settings for Route Policy Manager.

Table 39: Default route policy manager parameters

Parameters	Default
Route Policy Manager	Enabled
Administrative distance	115

Configure route policy manager

The route policy manager is a feature whose configuration commands in Cisco NX-OS may differ from those in Cisco IOS.

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

Configure 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 and create an IPv6 prefix list for IPv6.

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.

Beginning with Cisco NX-OS Release 9.3(9), make sure to add the sequence number when configuring the prefix-list in the NDFC/config-profile/dual-stage configuration modes. Also, when modifying a sequence or inserting a new one, ensure that there is a gap in the sequence number, preferably in increments of 5 or 10, instead of assigning a continuous number.

For example:

```
ip prefix-list allowprefix seq
10
permit 192.0.2.0/23 eq 24
ip prefix-list allowprefix seq
20
permit 209.165.201.0/27 eq 28
```

Note Beginning with Cisco NX-OS Release 9.3.9, if prefix-list does not have sequence numbers in the config-profile ensure to add the sequence numbers before upgrading to that release or higher.

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

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 Required: { **ip** | **ipv6** } **prefix-list** *name* **description** *string*

Example:

```
switch(config)# ip prefix-list
AllowPrefix description allows
engineering server
```

Adds an information string about the prefix list.

Step 3 { **ip** | **ipv6** } **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)# ipv6 prefix-list
AllowIPv6Prefix seq 10 permit 2001:0DB8:: le 32

switch(config)# ip prefix-list
even permit 0.0.0.0/32 mask 0.0.0.1

switch(config)# ipv6 prefix-list
even permit 2001:0DB8::/64 mask ffff:1::
```

Creates an IPv4 or IPv6 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. This option is available for IPv6 prefix lists beginning with Cisco NX-OS Release 9.3(3) for Cisco Nexus 9200, 9300-EX, and 9300-FX platform switches and 9700-EX and 9700-FX line cards .

Step 4 (Optional) **show** { **ip** | **ipv6** } **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(config)# copy running-config
startup-config
```

Saves this configuration change.

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 65535
switch(config-router)#
                neighbor 192.0.2.1/16 remote-as 65534
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 all /24 odd IP addresses:

```
switch# configure terminal
switch(config)# ip prefix-list list1 seq 7 permit 22.1.1.0/24 mask 255.255.1.0
switch(config)# show route-map test
route-map test, permit, sequence 7
Match clauses:
ip address prefix-lists: list1
Set clauses:
extcommunity COST:igp:10:20
switch(config)# show ip prefix-list list1
ip prefix-list list1: 1 entries
seq 7 permit 22.1.1.0/24 mask 255.255.1.0
```

This example shows how to create an IPv4 prefix list that matches all subnets of 21.1.0.0/16 where the subnet prefix is 17 or greater. Due to the mask option, only those incoming prefixes where the first bit in the third octet is unset (even) will be matched.

```
switch# configure terminal
switch(config)# ip prefix-list list1 seq 10 permit 21.1.0.0/16 ge 17 mask 255.255.1.0
```

Configure MAC lists

You can configure a MAC list to permit or deny a range of MAC addresses.

Beginning with Cisco NX-OS Release 10.4(2)F, make sure to add the sequence number when configuring the prefix-list in the NDFC/config-profile/dual-stage configuration modes. Also, when modifying a sequence or inserting a new one, ensure that there is a gap in the sequence number, preferably in increments of 5 or 10, instead of assigning a continuous number.

For example:

```
mac-list AllowMac seq
    5
permit 0022.5579.a4c1 ffff.ffff.0000
mac-list AllowMac seq
    10
permit 0033.5510.a4c1 ffff.ffff.0000
```



Note Beginning with Cisco NX-OS Release 10.4(2)F, if mac-list does not have sequence numbers in the config-profile ensure to add the sequence numbers before upgrading to that release or higher.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 Required: mac-list *name* seq *number* { permit | deny } *mac-address* [*mac-mask*]

Example:

```
switch(config)# mac-list AllowMac seq 5 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 name
```

Displays information about prefix lists.

Step 4 (Optional) **copy running-config startup-config**

Example:

```
switch(config)# copy running-config
                        startup-config
```

Saves this configuration change.

Configure 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, the permit or deny condition applies.



Note When you define an community-list regular expression in an NDFC or DCNM configuration profile, enclose the expression in double quotation marks. NX-OS stores the expression in quoted form. An equivalent unquoted expression in the profile can differ from the running configuration after a reload, causing profile application to fail because of conflicting commands.

Procedure

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.

Step 3 (Optional) **show** { **ip** | **ipv6** } **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.

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 65535:20
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
```

Replace BGP AS-path attribute

The Replace BGP AS-path attribute feature enables modification of the BGP as-path attribute in route maps to influence routing policy for eBGP neighbors.

- Applicable only to eBGP neighbors on a per address family identifier (AFI) basis; configuration on iBGP neighbors is ignored.
- Supports any combination of AS_SET, AS_SEQUENCE, CONFED_SET, and CONFED_SEQUENCE.
- Allows a maximum of 32 AS numbers in a **set as-path** or **set as-path replace** command.

Guidelines for replacing the BGP AS-path attribute

The following guidelines should be considered when replacing the BGP as-path attribute:

- A route map with this feature can be applied to both the inbound and outbound sides of a BGP neighbor.
- When interacting with a BGP speaker that supports only a 2-byte AS, the 4-byte AS number is replaced by the reserved 2-byte AS number 23456.
- If a confederation identifier is configured, use the confederation identifier as the local ASN in the CLI when interacting with a peer outside the confederation. For peers within the same confederation, use the process ASN in the **router bgp** *asn* command.
- When the BGP **local-as** feature is configured, the configured local-as will be considered as local ASN in the CLI.
- For outbound route-maps, the local ASN will always be prepended to the resulting as_path from the CLI.
- Only one of these options can be configured under one route-map sequence: **set as-path** , **set as-path prepend** , and **set as-path replace** .
- If **remove-private-as** is configured, it will be applied before applying the new route-map commands on the outbound side.
- If **as-override** is configured, it will be applied after applying the new route-map commands on the outbound side.
- AS_PATH loop checks execute on the original AS_PATH before the new route-map commands are applied on both inbound and outbound sides. These checks can be relaxed by using **allow-as in** on the inbound side and **disable-peer-as-check** on the outbound side.

Replace the complete AS-path

Use this procedure to modify the AS-path in an incoming or outgoing BGP update to a custom AS-path. You can also remove the AS-path completely.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global 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 [**no**] **set as-path** { **none** | { *as-number* | **remote-as** | **local-as** } +] }

Example:

```
switch(config-route-map)# set as-path 11 local-as remote-as 13
```

Replaces AS_PATH with a list of custom ASNs or clears the AS_PATH. The command options are:

- *as-number* : The specified AS number.
- **remote-as** : The AS number of the BGP peer.
- **local-as** : The local AS number.

The **none** keyword removes the AS-path completely.

In the following examples, these values are assumed:

- The original AS_PATH is **10 20 30 40 50 60** .
- The local-as is **100** .
- The remote-as is **200** .

This example shows how to specify a custom AS-path. This command will change the AS-path to **11 100 200 13 200 10.10 65535** .

```
switch# configure terminal
switch(config)# route-map Testmap permit 10
switch(config-route-map)# set as-path 11 local-as remote-as 13 remote-as 10.10 65535
```

This example shows how to clear the AS-path. This command will cause the AS-path to be empty.

```
switch# configure terminal
switch(config)# route-map Testmap permit 10
switch(config-route-map)# set as-path none
```

Replace selected AS numbers in the AS-path

Use this procedure to replace specific AS numbers in the AS-path and replace them with custom AS numbers in an incoming or outgoing BGP update. You can also specify **private-as** as a match keyword. In this case, any instance of a private-as is matched and can be replaced or removed.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global 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 [**no**] **set as-path replace** { *asn_list* | **private-as** } [**with** { *as-number* | **remote-as** | **none** }]

Example:

```
switch(config-route-map)# set as-path replace 1, 2, private-as with remote-as
```

If the **with** keyword is not specified, substitute the local-as for any instance of an ASN mentioned in the comma separated *asn_list* , or for any private-as if the **private-as** keyword is specified.

If the **with** keyword is specified, substitute the value after the **with** keyword for any matched ASN, or any private-as if the **private-as** keyword is specified.

The command options following the **with** keyword are:

- *as-number* : The matched values are replaced by the specified AS number.
- **remote-as** : The matched values are replaced by the AS number of the BGP peer.
- **none** : The matched values are removed from the AS-path.

In the following examples, these values are assumed:

- The original AS_PATH is **1 5 2 10.10 65534 20** .
- The local-as is **100** .
- The remote-as is **200** .

This example shows how to replace two specific ASNs and a private-as with the local-as. This command will change the AS-path to **100 5 100 10.10 100 20** .

```
switch# configure terminal
switch(config)# route-map Testmap permit 10
switch(config-route-map)# set as-path replace 1, 2, private-as
```

This example shows how to replace two specific ASNs and a private-as with the neighbor's ASN (remote-as). This command will change the AS-path to **200 5 200 10.10 200 20** .

```
switch# configure terminal
switch(config)# route-map Testmap permit 10
switch(config-route-map)# set as-path replace 1, 2, private-as with remote-as
```

This example shows how to remove two specific ASNs and a private-as. This command will change the AS-path to **5 10.10 20** .

```
switch# configure terminal
switch(config)# route-map Testmap permit 10
switch(config-route-map)# set as-path replace 1, 2, private-as with none
```

Configure 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.

Note When you define an community-list regular expression in an NDFC or DCNM configuration profile, enclose the expression in double quotation marks. NX-OS stores the expression in quoted form. An equivalent unquoted expression in the profile can differ from the running configuration after a reload, causing profile application to fail because of conflicting commands.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 Enter one of the following:

- **ip community-list standard** *list-name* { **deny** | **permit** } [*community-list*] [**internet**] [**local-AS**] [**no-advertise**] [**no-export**] [**graceful-shutdown**] [**blackhole**]

  or

- **ip community-list expanded** *list-name* { **deny** | **permit** } *expression*

Example:

```
switch(config)# ip community-list standard BGPCommunity permit no-advertise 65535:20
```

or

```
switch(config)# ip community-list expanded BGPComplex deny 50000:[0-9][0-9]
```

The first option 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.

The second option creates an expanded BGP community list using a regular expression.

Step 3 (Optional) **show ip community list** *name*

Example:

```
switch(config)# show ip community-list BGPCommunity
```

Displays information about community lists.

Step 4 (Optional) **copy running-config startup-config**

Example:

```
switch(config)# copy running-config startup-config
```

Saves this configuration change.

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 65535:20
switch(config)# ip community-list standard BGPCommunity permit local-AS no-export
switch(config)# copy running-config startup-config
```

Configure extended community lists

You can use extended community lists to filter BGP routes based on the community attribute. The community number consists of a 6-byte value in the *aa4:nn* format. The first four 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 extended community list statement, all extended community values must match to satisfy the extended community list filter. When you configure multiple values in separate extended community list statements, the first list that matches a condition is processed.

Use extended community lists in a match statement to filter BGP routes based on the extended community attribute.

Note Configure **extcommunity** in AS2:NN or AS4:NN (as-plain) formats always.

Beginning with NX-OS release 10.4(3)F, you can configure **extcommunity** in AS.dot format.

Note When you define an community-list regular expression in an NDFC or DCNM configuration profile, enclose the expression in double quotation marks. NX-OS stores the expression in quoted form. An equivalent unquoted expression in the profile can differ from the running configuration after a reload, causing profile application to fail because of conflicting commands.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 Enter one of the following:

- **ip extcommunity-list standard** *list-name* { **deny** | **permit** } **seq** *5* **4byteas-generic** { **transitive** | **nontransitive** } *community1* [*community2...*] **rt** *2:2* **soo** *3:3*

  or

- **ip extcommunity-list expanded** *list-name* **seq** *5* { **deny** | **permit** } *expression*

Example:

```
switch(config)# ip extcommunity-list
standard BGPExtCommunity seq 5 permit
4byteas-generic transitive 65535:20 rt 2:2 soo 3:3
```

or

```
switch(config)# ip extcommunity-list
expanded BGPExtComplex seq 5 deny
1.5:[0-9][0-9]
```

The first option creates a standard BGP extended community list. The *community* can be one or more extended communities in the *aa4:nn* format.

The second option creates an expanded BGP extended community list using a regular expression.

Step 3 **ip extcommunity-list standard** *commext* **seq** *5* **permit 4byteas-generic transitive** *1:1* **rt** *2:2* **soo** *3:3*

Sequence number is added as an input parameter to the CLI.

Henceforth, you must enter the input sequence number while configuring extcommunity lists.

Example:

```
switch(config)# ip extcommunity-list standard commext seq 5 permit 4byteas-generic transitive 1:1 rt
 2:2 soo 3:3
```



Note For config replace, the user config file must contain a valid running configuration collected from a device. It can be collected from a device running any NX-OS image label. It must be a valid file that which is not tampered manually.

Step 4 (Optional) **show ip community-list** *name*

Example:

```
switch(config)# show ip community-list
BGPCommunity
```

Displays information about extended community lists.

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

Example:

```
switch(config)# copy running-config
startup-config
```

Saves this configuration change.

This example shows how to create a generic specific extended community list:

```
switch# configure terminal
switch(config)# ip extcommunity-list standard test1 seq 5 permit 4byteas-generic transitive
65535:40 65535:60
switch(config)# copy running-config startup-config
```

Configure route maps

Configuring a route map for BGP triggers an automatic soft clear or refresh of BGP neighbor sessions.

Use this block to include any additional information that helps orient the reader to the task, aiding in successful task completion.

Before you begin

Follow these steps to configure route maps.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global 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(config-route-map)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

You can configure the following optional match parameters for route maps in route-map configuration mode:

You can configure the following optional match parameters for route maps in route-map configuration mode:

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.

Command	Purpose
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. **Note** BGP does not support this command.

Global commands to block the deletion of route-map

The global commands are by default generic. Beginning with Cisco NX-OS release 10.2(2)F, the functionality to block the route-map deletion, if used by client is applicable only for BGP.

- Use the **system default route-map validate-applied** command to enable the blocking of the deletion of route-map.
- Use the **no system default route-map validate-applied** command to disable the blocking of the deletion of route-map.
- When you configure **system default route-map validate-applied**, deletion of any route-map is blocked in these conditions:
  - When this command is enabled in a running configuration
  - The **route-map** <> is enabled in running configuration, and is also referenced by an active BGP client.

  These restrictions are imposed to protect users from accidently deleting an active route-map.
- Use the **show running-config rpm** command to view the non-default configuration.
- Use the **show running-config rpm all** command to view the default configuration.
- By default this command is in default state.

Verify the route policy manager configuration

To display 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** | **ipv6**] **prefix-list** [*name*]	Displays information about an IPv4 or IPv6 prefix list.
show route-map [*name*]	Displays information about a route map.
show route-map [*name*] **brief**	Provides information about blocking route-map deletion functionality and the list of clients associated with the route-map.

Configuration examples for route policy manager

This topic provides configuration examples for creating IPv4 prefix lists, AS-path lists, and using address families with Route Policy Manager.

This example shows how to use an address family to configure Route Policy Manager so that any unicast and multicast routes from neighbor 172.16.0.1 are accepted if they match prefix-list AllowPrefix:

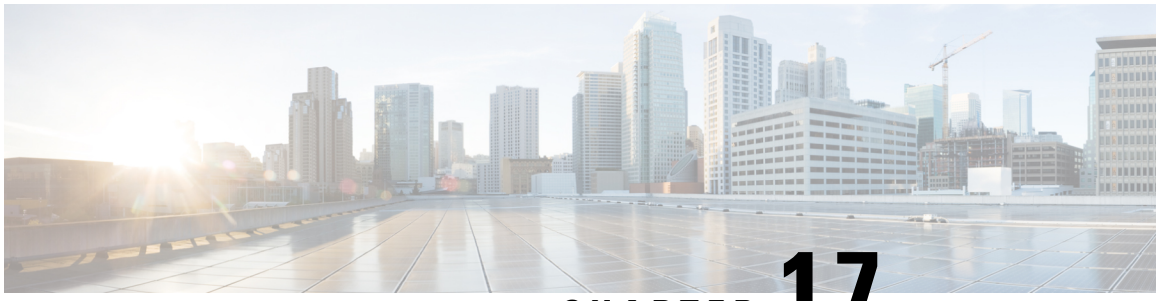
```
router bgp 64496
neighbor 172.16.0.1 remote-as 64497
address-family ipv4 unicast
route-map filterBGP in
route-map filterBGP
match ip address prefix-list AllowPrefix
ip prefix-list AllowPrefix 10 permit 192.0.2.0/24
ip prefix-list AllowPrefix 20 permit 172.16.201.0/27
```

Related topics

The following topics can give more information on Route Policy Manager.

- Related topic: *Configuring Basic BGP*

Related topics



CHAPTER 17

Configuring Policy-Based Routing

This chapter contains the following sections:

- About policy-based routing, on page 555
- Prerequisites for policy-based routing, on page 559
- Guidelines and limitations for policy-based routing, on page 559
- Default settings for policy-based routing, on page 563
- Configure policy-based routing, on page 563
- Verify the policy-based routing configuration, on page 576
- Configuration examples for policy-based routing, on page 576
- Related documents for policy-based routing, on page 579

About policy-based routing

Policy-based routing is a method that enables the configuration of defined policies for IPv4 and IPv6 traffic flows, allowing for more granular control over packet forwarding decisions beyond standard routing protocols.

- All packets received on an interface with policy-based routing enabled are processed through enhanced packet filters or route maps.
- Route maps dictate the policy that determines where to forward packets.
- Policy-based routing lessens the reliance on routes derived from routing protocols.

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 9000 Series NX-OS Quality of Service Configuration Guide).
- Load sharing—Distributes traffic among multiple paths based on the traffic characteristics.

Policy route maps

A policy route map is a set of entries, each containing match and set statements that determine how packets are routed based on defined criteria.

- Match statements specify the conditions that packets must meet for a policy to apply.
- Set clauses define the routing actions to be taken when the match criteria are met.
- Each route-map statement can be marked as permit or deny, affecting how packets are processed.

How policy route maps process packets

Policy route maps use permit and deny statements to control packet routing based on match criteria.

- If a statement is marked as **permit** and the packets meet the match criteria, the set clause is applied, such as choosing the next hop.
- If a statement is marked as **deny** , packets that meet the match criteria are sent back through normal forwarding channels and destination-based routing is performed.
- If a statement is marked as **permit** and packets do not match any route-map statements, the packets are sent back through 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 **set** criteria for policy-based routing define how packets are forwarded by specifying next-hop addresses, VRFs, or interfaces in route maps.

- Commands are mutually exclusive within a route-map sequence.
- The first command uses the IP address of the adjacent next-hop router to forward packets.
- If no match criteria are met, packets are routed through the normal destination-based process.

The following **set** commands are supported for route maps in policy-based routing:

- **set** { **ip** | **ipv6** } **next-hop**
- **set** { **ip** | **ipv6** } **default next-hop**
- **set** { **ip** | **ipv6** } **vrf** *vrf-name* **next-hop**
- **set** { **ip** | **ipv6** } **default vrf** *vrf-name* **next-hop**
- **set interface null0**
- **set vrf** *vrf-name*

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.

For more information on set commands configuration, see Configure a route policy, on page 566 section.

Route map support matrix for policy-based routing

The following table includes the configurable match and set statements for policy-based routing on Cisco Nexus 9000 Series Switches running the latest shipping release.

The following legend applies to the tables:

- Yes—The statement is supported for policy-based routing.
- No—The statement is not supported for policy-based routing.
- If a statement does not apply for policy-based routing, there is an em dash (—) in the column next to the statement.
- Where clarification is required, information is added in the appropriate row/column.

Table 40: SET route map statements for policy-based routing

SET route map statement	Policy-based routing (PBR)
IPv4 Next Hop	Yes
IPv6 Next Hop	Yes
IPv4 vrf Next Hop	Yes
IPv6 vrf Next Hop	Yes
Default IPv4 Next Hop	Yes
Default IPv6 Next Hop	Yes
Default IPv4 vrf Next Hop	Yes
Default IPv6 vrf Next Hop	Yes
IPv4 Next Hop Verify Availability	Yes
IPv6 Next Hop Verify Availability	Yes
IPv4 vrf Next Hop Verify Availability	Yes

SET route map statement	Policy-based routing (PBR)
IPv6 vrf Next Hop Verify Availability	Yes
Default IPv4 Next Hop Verify Availability	Yes
Default IPv6 Next Hop Verify Availability	Yes
Default IPv4 vrf Next Hop Verify Availability	Yes
Default IPv6 vrf Next Hop Verify Availability	Yes
Interface null0	Yes
VRF	Yes

Route-map processing logic

Route-map processing logic defines how packets are evaluated and routed based on the sequence and type of route-map statements (permit or deny), the presence of match and set commands, and the status of next-hop addresses.

- Each route-map statement is processed in sequence when a packet is received on an interface with a route map.
- If a statement is **permit** , the packet is matched against the **match** command criteria, and if matched, the **set** command action is executed.
- If a statement is **deny** , the packet is matched against the **match** command criteria, and if matched, policy-based routing stops and the default routing table is used.

Route-map statement processing and next-hop handling

When an interface with a route map receives a packet, the forwarding logic processes each route-map statement according to the sequence number.

- 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.

1. If the route-map statement encountered is a **route-map... permit** statement, the packet is matched against the criteria in the **match** command. If the packet matches the permit ACEs in the ACL, the policy-based routing logic executes the action that the **set** command specifies on the packet.
2. If the route-map statement encountered is a **route-map... deny** statement, the packet is matched against the criteria in the **match** command. If the packet matches the permit ACEs in the ACL, policy-based routing processing stops, and the packet is routed using the default IP routing table.

Note The **set** command has no effect inside a **route-map... deny** statement.

Beginning Cisco NX-OS Release 9.2(3), you can balance policy-based routing traffic if the next hop is recursive over ECMP paths using the **next-hop ip-address load-share** command. This situation is supported on the following switches, line cards, and modules:

- N9K-C9504 switches with -R line cards
- N9K-C9508 switches with -R line cards
- N9K-C9372TX
- N9K-X9564TX
- N9K-X9732C-EX

For all the next hop routing requests, the Routing Profile Manager (RPM) resolves them using unicast Routing Information Base (uRIB). RPM also programs all ECMP paths, which helps to uniformly load balance all the ECMP paths. PBR over ECMP is supported only on IPv4.

Prerequisites for policy-based routing

Policy-based routing requires specific prerequisites to be met before configuration.

- Install the correct license.
- You must enable policy-based routing.
- Assign an IP address on the interface and bring the interface up before you apply a route map on the interface for policy-based routing.

Guidelines and limitations for policy-based routing

The configuration guidelines and limitations for policy-based routing (PBR) on Cisco Nexus 9000 Series switches are listed below.

- Policy-based routing (PBR) is subject to specific configuration guidelines and limitations depending on the Cisco Nexus switch platform and software release.
- Support for IPv4 and IPv6 PBR, route-map statements, ACL usage, and feature compatibility varies by platform and line card.
- Some features, such as PBR over VXLAN, fast convergence, and set vrf, are only available on certain platforms or software versions.

Policy-based routing has the following configuration guidelines and limitations:

- Cisco Nexus 9500 platform switches with 9700-EX/FX line cards do not support PBR IPv6 Default Next hop for FIB Miss traffic.

- The following switches support IPv4 and IPv6 policy-based routing:
  - Cisco Nexus 9200 platform switches
  - Cisco Nexus 9300-EX/FX/FX2/FX3 /GX /H1/H2R /GX2 platform switches.
  - Cisco Nexus 9508 switches with 9636C-R, 9636C-RX, and 9636Q-R line cards (For these line cards, PBR policy has a higher priority over attached and local routes. Explicit white 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 policy-based routing route map can have only one set statement per route-map statement, unless you are using IP SLA policy-based routing. For information on IP SLA policy-based routing, see the *Cisco Nexus 9000 Series NX-OS IP SLAs Configuration Guide* .



Note Cisco Nexus 9508 switches with 9636C-R, 9636C-RX, and 9636Q-R line cards do not support IP SLA.

- 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 not supported with inbound traffic on FEX ports.
- Policy-based routing is not supported on FEX ports for Cisco Nexus 9300-EX platform switches.
- Only Cisco Nexus 9508 switches with 9636C-R, 9636C-RX, and 9636Q-R line cards support policy-based routing with Layer 3 port-channel subinterfaces.
- Beginning with Cisco NX-OS Release 10.1(2), policy-based routing with Layer 3 port-channel subinterfaces are supported on Cisco Nexus 9300-X Cloud Scale Switches.
- 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.
- When you configure multiple features on an interface (such as PBR and ingress ACL), the ACLs for those features are merged for TCAM optimization. As a result, statistics are not supported.
- Enabling the vPC peer-gateway configuration allows a vPC switch to locally route packets destined for its peer's MAC address. Since these packets are processed at Layer 3, any configured PBR (Policy-Based Routing) or Layer 3 ACLs are applied to this traffic. This is an expected behaviour.
- For PBR with VXLAN, the load-share keyword is not required.

Note Cisco Nexus 9500 platform switches with the 9700-EX/FX line cards support IPv4/IPv6 policy-based routing over VXLAN. Cisco Nexus 9508 switches with 9636C-R, 9636C-RX, and 9636Q-R line cards do not support policy-based routing over VXLAN.

- The Cisco Nexus 9000 Series switches support policy-based ACLs (PBACLs), also referred to as object-group ACLs. For more information, see the *Cisco Nexus 9000 Series NX-OS Security Configuration Guide* .

Note Cisco Nexus 9508 switches with 9636C-R, 9636C-RX, and 9636Q-R line cards do not support PBACLs.

- The following guidelines and limitations apply to PBR over VXLAN EVPN:
  - PBR over VXLAN EVPN is supported only for Cisco Nexus 9300-EX/FX/FX2/ FX3/ GX/GX2 platform switches.
  - PBR over VXLAN EVPN does not support the following features: VTEP ECMP and the load-share keyword in the **set {ip | ipv6} next-hop** *ip-address* command.
  - PBR over VXLAN EVPN support **set {ip | ipv6} vrf** *vrf-name* **next-hop** *ip-address* command, and by using multiple lines of **set {ip | ipv6} vrf** *vrf-name* **next-hop** *ip-address* command PBR over VXLAN EVPN supports different VRF on each multiple next-hop.
- The following guidelines and limitations apply to PBR over tunnel interface:
  - Beginning with Cisco NX-OS Release 10.3(3)F, the PBR next-hop redirecting to a tunnel interface is supported on Cisco Nexus 9000 Series platform switches with the following limitations:
    - Only **gre ip** and **ipip ip** modes are supported.
    - The **load-share** keyword in the route-map, won't be supported if multiple configured next-hops resolve to combination of tunnel interface and non-tunnel interface.
    - Overlay ECMP (same next-hop resolving to multiple tunnels with equal cost path) is not supported.
- The following guidelines and limitations apply to PBR fast convergence:
  - PBR fast convergence is supported only for policies that have route-map sequences defined with multiple alternate next-hops, without load-share option, and with SLA probes for tracking next-hop availability.
  - Simultaneous failures of primary and back-up next-hops are not handled in the fast path. In such events, the system will fall back to control plane updates.
  - PBR fast convergence is primarily supported in events where adjacency loss is detected.
  - PBR fast convergence is not supported for next-hops reachable over VXLAN.

- PBR fast convergence should not be used when next-hops are specified with millisecond SLAs/tracks to track availability.

  For more information about SLA, see the *Cisco Nexus 9000 Series NX-OS IP SLAs Configuration Guide* .

- When PBR fast convergence is disabled, the number of ACL redirect entries is proportional to the number of unique primary next-hops across the PBR policies. When PBR fast convergence is enabled, the system may require ACL redirect entries per port-slice that is proportional to the number of unique combinations of primary and back-up next-hops configured across the route-map sequences in the PBR policies.

- The following platforms support PBR fast convergence: Cisco Nexus 9300-FX/FX2/FX3/GX/GX2/H1/H2R platform switches.

- Beginning with Cisco NX-OS Release 10.3(2)F, default IPv4/IPv6 next-hop VRF selection for PBR is provided on Cisco Nexus 9000 Series platform switches.
- Beginning with Cisco NX-OS Release 10.3(2)F, PBR over IP Tunnels is supported only for tunnels having gre and ipip mode. However, PBR over IP Tunnels does not support the **load-share** keyword in all variants of **set {ip | ipv6} next-hop** commands.
- The following guidelines and limitations apply to PBR Set VRF:
  - Beginning with Cisco NX-OS Release 10.5(2)F, The **set vrf** and **pbr set-vrf recirculation interface** *port-channel-num* commands are supported on Cisco Nexus 9300-FX2, FX3, GX, GX2, H2R, H1 Series switches, 9500 Series switches with 9700 GX line cards, and N9K-C9504-FM-G or N9K-C9508-FM-G fabric module. However, mixed mode is not supported on EoR.
  - Next-hops resolving to tunnels like VXLAN endpoints, GRE, IP-in-IP tunnels, and so on, are not supported.
- Beginning with Cisco NX-OS Release 10.5(3)F, PBR now supports Layer 3 subinterfaces, routed interfaces, and port-channel subinterfaces. This expands the types of interfaces usable with PBR policies
- Beginning with Cisco NX-OS Release 10.6(1)F, the following features of policy-based routing are supported on Cisco Nexus N9336C-SE1 switches:
  - Base redirection
  - 32 way ECMP
  - NULL routes
  - IPv4 and IPv6 PBR policies

  And the following features are not supported:
  - Default next hop
  - Set VRF
  - Fast convergence
  - MPLS
  - SRTE

- VXLAN
- Tunnel support

- Beginning with Cisco NX-OS Release 10.6(1)F, PBR is supported on Cisco Nexus N9324C-SE1U and N9348Y2C6D-SE1U switches.

Default settings for policy-based routing

This topic provides the default settings for policy-based routing.

Table 41: Default Policy-Based Routing Parameters

Parameters	Default
Policy-based routing	Disabled

Configure policy-based routing

Enable the policy-based routing feature

You must enable the policy-based routing feature before you can configure a route policy.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

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.

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
```

Copies the running configuration to the startup configuration.

Enable the policy-based routing over ECMP

PBR over ECMP is not enabled by default. You must enable the policy-based routing feature before you can configure a route policy.

Procedure

Step 1 Use the **configure terminal** command to enter global configuration mode.

Example:

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

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.

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 **hardware profile pbr ecmp paths** <*maxpath*> | **[no] hardware profile pbr ecmp paths** [<*maxpath*>]

Example:

```
switch(config)# hardware profile pbr ecmp paths 12
                        Warning!!: The pbr ecmp path limits have been changed.
```

```
                        Please reload the switch now for the change to take effect.
                        switch(config)#

switch(config)# no hardware profile pbr ecmp paths 12
                        Warning!!: The pbr ecmp path limits have been changed.
                        Please reload the switch now for the change to take effect.
                        switch(config)#

switch(config)# no hardware profile pbr ecmp paths
                        Warning!!: The pbr ecmp path limits have been changed.
                        Please reload the switch now for the change to take effect.
                        switch(config)#
```

Configure the number of ECMP paths for IP next hop. However, the traffic may not go through all the paths unless you explicitly configure the load share in the set IP next hop. Whenever you remove or modify the PBR ECMP paths, the changes will take effect only after next reload. The range is from 1 through 64.

Step 5 **show system internal rpm state**

Displays the currently configured and operational values of PBR ECMP paths.

Configure PBR fast convergence

Configure PBR fast convergence to reduce traffic convergence time to sub-second in the event of a next-hop failure.

In the case of a failure of a next-hop that is currently in use in PBR, PBR fast convergence can reduce the traffic convergence time to sub-second. PBR fast convergence assists policies that have route-map sequences defined with multiple alternate next-hops, without the load-share option, and with SLA probes for tracking next-hop availability.

PBR fast convergence is disabled on the switch by default. After configuring PBR fast convergence and saving the configuration, you must reload the switch to activate PBR fast convergence.

Before you begin

You must enable the policy-based routing feature before you can configure PBR fast convergence.

Procedure

Step 1 **configure terminal**

Example:

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

Use this command to enter global configuration mode.

Step 2 [**no**] **feature pbr**

Example:

```
switch(config)# feature pbr
```

Enables the policy-based routing feature.

Step 3 [**no**] **hardware profile pbr next-hop fast-convergence**

Example:

```
switch(config)# hardware profile pbr next-hop fast-convergence
```

Configures PBR fast convergence.

Use the **no** form of this command to disable PBR fast convergence.

Note Enabling or disabling PBR fast convergence takes effect after the switch is reloaded.

Step 4 **copy running-config startup-config**

Example:

```
switch(config)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

This example enables PBR fast convergence and reloads the switch:

```
switch(config)# hardware profile pbr next-hop fast-convergence
Warning: Please save config and reload the system for the configuration to take effect.
switch(config)# copy running-config startup-config
switch(config)# reload
```

What to do next

After enabling or disabling PBR fast convergence and saving the configuration, reload the switch.

Configure a route policy

You can use route maps in policy-based routing to assign routing policies to the inbound interface. Cisco NX-OS routes the packets when it finds a next hop and an interface.

Procedure

Step 1 **configure terminal**

Example:

```
switch#
configure terminal
```

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 policy route-map** *map-name*

Example:

```
                        switch(config-if)#
                        ip policy route-map Testmap
switch(config-route-map)#
```

Assigns a route map for IPv4 policy-based routing to the interface.

Step 4 **ipv6 policy route-map** *map-name*

Example:

```
                        switch(config-if)#
                        ipv6 policy route-map Testmap
switch(config-route-map)#
```

Assigns a route map for IPv6 policy-based routing to the interface.

Step 5 **match** { **ip** | **ipv6** } **address [accesslist-name]**

Example:

For IPv4

```
switch(config-route-map)# match ip
address ACL1_v4
```

For IPv6

```
switch(config-route-map)# match ipv6
address ACL1_v6
```

Matches an IPv4 or IPv6 address against one or more IP or IPv6 access control lists (ACLs). This command is used for policy-based routing and is ignored by route filtering or redistribution.

Step 6 **set** { **ip** | **ipv6** } **next-hop** *address1* [*address2...*] **[load-share] [drop-on-fail] [force-order]**

Example:

For IPv4

```
switch(config-route-map)# set ip next-hop 192.0.2.1
```

For IPv6

```
switch(config-route-map)# set ipv6 next-hop 2001:0DB8::1
```

Sets the IPv4 or 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.

Use the optional **force-order** keyword to enable next-hop ordering as specified in the CLI.

Use the optional **drop-on-fail** keyword to drop packets instead of using default routing when the configured next hop becomes unreachable. This option is supported for Cisco Nexus 9200, 9300-EX/FX/FX2 and 9364C platform switches and Cisco Nexus 9500 platform switches with -EX/FX line cards.

Step 7 **set** { **ip** | **ipv6** } **vrf** *vrf-name* **next-hop** *address1* [*address2...*] **[force-order] [drop-on-fail] [load-share]**

Example:

For IPv4

```
switch(config-route-map)#
set ip vrf vrf1 next-hop 192.0.2.2
```

For IPv6

```
switch(config-route-map)#
set ipv6 vrf vrf1 next-hop 2001:0DB8::1
```

Sets the IPv4 or IPv6 next-hop address based on default or user-defined vrf for policy-based routing.

This command supports inter-VRF routing packets arriving at a VRF interface are routed through any other VRF based on configured next-hop.

This command uses the first valid next-hop address if multiple addresses are configured.

Use the optional **force-order** keyword to enable next-hop ordering as specified in the CLI.

Use the optional **drop-on-fail** keyword to drop packets instead of using default routing when the configured next hop becomes unreachable. This option is supported for Cisco Nexus 9200, 9300-EX/FX/FX2 and 9364C platform switches and Cisco Nexus 9500 platform switches with -EX/FX line cards.

Use the optional **load-share** keyword to load balance traffic across a maximum of 32 next-hop addresses.

Step 8 **set** { **ip** | **ipv6** } **default next-hop** *address2* [*address2...*] [**load-share**]

Example:

For IPv4

```
switch(config-route-map)#
set ip default next-hop 192.0.2.2
```

For IPv6

```
switch(config-route-map)#
set ipv6 default next-hop 2001:0DB8::1
```

Sets the IPv4 or IPv6 next-hop address for policy-based routing when there is no explicit route to a destination. This command uses the first valid next-hop address if multiple addresses are configured. This can done with next-hop tracking only.

- Use the optional **load-share** keyword to load balance traffic across a maximum of 32 next-hop addresses.

From Cisco NX-OS Release 10.2(2)F, below are supported:

- Command **set ip default next-hop** is supported on GX, GX2, and FX3 platform switches.
- Use the optional **verify-availability** keyword to verify the reachability of the tracked object.



Note This command is currently not supported on N9K-C950x.

Step 9 **set** { **ip** | **ipv6** } **default vrf** *vrf-name* **next-hop** *address1* [*address2...*] **[load-share]**

Example:

For IPv4

```
switch(config-route-map)#
set ip default vrf vrf1 next-hop 192.0.2.2
```

For IPv6

```
switch(config-route-map)#
set ipv6 default vrf vrf1 next-hop 2001:0DB8::1
```

Sets the IPv4 or IPv6 next-hop address for policy-based routing when there is no explicit route to a destination.

This command supports inter-VRF routing packets arriving at a VRF interface that are routed through any other VRF based on configured next-hop.

This command uses the first valid next-hop address if multiple addresses are configured.



Note This command does not allow multiple VRFs in set statement.

Use the optional **load-share** keyword to load balance traffic across a maximum of 32 next-hop addresses.

Step 10 **set** { **ip** | **ipv6** } **next-hop verify-availability** *next-hop-address* **track** *object*

Example:

```
switch(config-route-map)# set ip next-hop verify-availability 192.0.2.2 track 1
```

Sets the IPv4 or IPv6 next-hop address for policy-based routing.

Use this command to configure policy routing to verify the reachability of the next hop of the route map before the switch performs policy routing to that next hop. Repeat this step to configure the route map to verify the reachability of other tracked objects.



Note For additional information about object tracking,see the Cisco Nexus 9000 Series NX-OS IP SLAs Configuration Guide.

Step 11 **set** { **ip** | **ipv6** } **vrf** *vrf-name* **next-hop verify-availability** *next-hop-address* **track** *object*

Example:

```
switch(config-route-map)# set ip vrf vrf1 next-hop verify-availability 192.0.2.2 track 1
```

Sets the IPv4 or IPv6 next-hop address based on default or user-defined vrf for policy-based routing.

This command supports inter-VRF routing packets arriving at a VRF interface are routed through any other VRF based on configured next-hop.

Use this command to configure policy routing to verify the reachability of the next hop of the route map before the switch performs policy routing to that next hop. Repeat this step to configure the route map to verify the reachability of other tracked objects.



Note For additional information about object tracking, see the Cisco Nexus 9000 Series NX-OS IP SLAs Configuration Guide.

Step 12 **set** { **ip** | **ipv6** } **default next-hop verify-availability** *next-hop-address* **track** *object*

Example:

```
switch(config-route-map)# set ip default next-hop verify-availability 192.0.2.2 track 1
```

Sets the IPv4 or IPv6 next-hop address for policy-based routing when there is no explicit route to a destination.

Use this command to configure policy routing to verify the reachability of the next hop of the route map before the switch performs policy routing to that next hop. Repeat this step to configure the route map to verify the reachability of other tracked objects.



Note For additional information about object tracking,see the Cisco Nexus 9000 Series NX-OS IP SLAs Configuration Guide.

Step 13 **set** { **ip** | **ipv6** } **default vrf** *vrf-name* **next-hop verify-availability** *next-hop-address* **track** *object*

Example:

```
switch(config-route-map)# set ip default vrf vrf1 next-hop verify-availability 192.0.2.2 track 1
```

Sets the IPv4 or IPv6 next-hop address for policy-based routing when there is no explicit route to a destination.

This command supports inter-VRF routing packets arriving at a VRF interface that are routed through any other VRF based on configured next-hop.

Use this command to configure policy routing to verify the reachability of the default VRF next hop of the route map before the switch performs policy routing to that next hop. Repeat this step to configure the route map to verify the reachability of other tracked objects.



Note For additional information about object tracking, see the Cisco Nexus 9000 Series NX-OS IP SLAs Configuration Guide.

Step 14 **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 15 **set vrf** *vrf-name*

Example:

```
switch(config-route-map)#
set vrf vrf1
```

The **set vrf** command in a route-map allows you to specify the VRF context that should be used for route lookups or next-hop resolution.



Note There are some prerequisite configurations that are required for the **set vrf** command to work. For more information, see Redirect to VRF Using a Route Map, on page 571 section.

Redirect to VRF Using a Route Map

The **set vrf** command in a route-map allows you to specify the VRF context that should be used for route lookups or next-hop resolution. This command allows you to control which VRF is used to lookup routes, rather than defaulting to the VRF associated with the ingress interface.

The set VRF feature is supported through a two-pass solution. This involves using a dedicated link-loopback interface to redirect packets into the desired VRF.

- In Pass-1, the packets ingressing PBR interface after route-map evaluation is redirected to a dedicated portchannel dot1q interface which has the member-ports configured in link-loopback mode. The port-channel dot1q interface should be member of the desired redirect VRF.
- In Pass-2, due to the link-loopback configuration of the member-ports, the packets are re-circulated back to the same dedicated port-channel dot1q interface and default routing happens on the desired VRF.

The **pbr set-vrf recirculation interface port-channel** command is used to configure VRF to recirculation port-channel sub-interface mapping.

Procedure

Step 1 **configure terminal**

Example:

```
switch# configure terminal
```

Enters global configuration mode.

Step 2 **vrf context***vrf-name*

Example:

```
switch(config)# vrf context vrf1
                        switch(config-if)#
```

Enters VRF configuration mode.

Note This step is not required if the recirculation interface is configured for default VRF.

Step 3 **pbr set-vrf recirculation interface port-channel***port-channel-num*

Example:

```
switch(config-if)# pbr set-vrf recirculation interface port-channel 1.1
```

Configures set-vrf recirculation interface under VRF context where port channel is used to redirect packet to the VRF.

Configure Set VRF

For the **set vrf** command to function, follow these steps:

1. Enter global configuration mode.

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

2. Enable feature PBR.

```
switch(config)# feature pbr
```

3. Create a dedicated port-channel interface.

```
switch(config)# interface port-channel1
                              switch(config-if)# mtu 9216
                              switch(config-if)# no shutdown
```

4. Enable link-loopback configuration on an Ethernet interface and assign it to the channel group.

```
switch# configure terminal
                              switch(config)# interface Ethernet1/1
                              switch(config-if)# mtu 9216
                              switch(config-if)# link loopback
                              switch(config-if)# channel-group 1
                              switch(config-if)# no shutdown
```

5. Create a port-channel sub-interface with the configurations that are required for packet forwarding to the desired VRF.

```
switch# configure terminal
                              switch(config)# interface port-channel1.1
                              switch(config-if)# encapsulation dot1q 101
                              switch(config-if)# vrf member vrf1
                              switch(config-if)# ip forward
                              switch(config-if)# ipv6 address use-link-local-only
                              switch(config-if)# ipv6 nd dad attempts 0
                              switch(config-if)# ipv6 nd prefix default no-advertise
```

```
                                switch(config-if)# ipv6 nd suppress-ra
                                switch(config-if)# no shutdown
```

6. Define VRF to recirculation port-channel sub-interface mapping.

```
switch# configure terminal
                                switch(config)# vrf context vrf1
                                switch(config-vrf)# pbr set-vrf recirculation interface
port-channel1.1
```

7. Configure a route-map with **set vrf***vrf-name* command.

```
switch# configure terminal
                                switch(config)# route-map test permit 10
                                switch(config-route-map)# match ip address acl1
                                switch(config-route-map)# set vrf vrf1
                                switch(config-route-map)# route-map testv6 permit 10
                                switch(config-route-map)# match ip address acl1_v6
                                switch(config-route-map)# set vrf vrf1
```

8. Apply policy to the ingress interface.

```
switch# configure terminal
                                switch(config)# interface eth1/10
                                switch(config-if)# ip policy route-map test
                                switch(config-if)# ip policy route-map testv6
```

Note

- Link-loopback interface, port-channel and port-channel sub-interfaces configured for set vrf feature must not be utilized for any other purpose.
- If additional bandwidth is needed, an additional member port can be added to the channel group.
- Link-loopback configuration must be enabled explicitly on each member ports.
- A new port-channel sub-interface can be created within the same channel-group or a different one for various VRF redirection using the above steps.
- Each port-channel recirculation sub interface must have unique dot1q label.
- Ensure that the VRF member assigned to the port-channel recirculation sub-interface matches the VRF context under which the pbr set-vrf recirculation interface is configured.
- While using the pbr set vrf feature, there could be a slight increase in packet latency because of two pass solution.
- Ensure that all the member ports in a channel are configured with same port speed.
- If IPv6 related configurations are made in recirculation PO sub-interface, the following syslog messages are generated:

```
2024 Jun 17 22:08:07 Leaf-switch %ICMPV6-3-ND_LOG: icmpv6 [16250] Own mac address
ecce.13e2.271f in the NA packet received on port-channel1.1 from fe80::eece:13ff:fee2:271f

                                2024 Jun 17 22:56:58 Leaf-switch %ICMPV6-3-ND_LOG:
icmpv6 [16250] Duplicate target address=fe80::eece:13ff:fee2:271f detected on
interface=port-channel1.1 in NS packet
```

Redirect default route match to next-hop

Beginning with Cisco NX-OS Release 10.3(3)F, you can redirect the default route match to next-hop on Cisco Nexus 9300-EX/FX/FX2/GX platform switches.

Procedure

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.

Step 3 **hardware access-list tcam pbr match-default-route**

Example:

```
switch(config)# hardware access-list tcam pbr match-default-route
```

Redirects the packets that match the default route to a specified Next-Hop in the policy.

When the **hardware access-list tcam pbr match-default-route** command is used, the following order is followed during traffic forwarding:

```
Specific FIB route => PBR => Default route Explanation – Specific route will be
preferred over PBR 2)
```

Note When the command is enabled, it will take effect on all the new polices configured.

If this command is not enabled, the following order is followed during traffic forwarding:

```
Any FIB route (specific route or default route) => PBR Explanation – Any route
(specific route or default route ) will be preferred over PBR 3)
```

Step 4 { **ip** | **ipv6** } **policy route-map** *map-name*

Example:

For IPv4

```
                        switch(config-if)#
                        ip policy route-map Testmap
```

For IPv6

```
switch(config-if)#
ipv6 policy route-map Testmap
```

Assigns a route map for IPv4/IPv6 policy-based routing to the interface.

Step 5 **route-map** *map-name*

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.

Step 6 **match** { **ip** | **ipv6** } **address [accesslist-name]**

Example:

For IPv4

```
switch(config-route-map)# match ip
                        address ACL1_v4
```

For IPv6

```
switch(config-route-map)# match ipv6
                        address ACL1_v6
```

Matches an IPv4 or IPv6 address against one or more IP or IPv6 access control lists (ACLs). This command is used for policy-based routing and is ignored by route filtering or redistribution.

Step 7 **set** { **ip** | **ipv6** } **default next-hop** *address2* [*address2...*] [**load-share**]

Example:

For IPv4

```
switch(config-route-map)#
set ip default next-hop 192.0.2.2
```

For IPv6

```
switch(config-route-map)#
set ipv6 default next-hop 2001:0DB8::1
```

Sets the IPv4 or IPv6 next-hop address for policy-based routing when there is no explicit route to a destination. This command uses the first valid next-hop address if multiple addresses are configured. This can done with next-hop tracking only.

- Use the optional **load-share** keyword to load balance traffic across a maximum of 32 next-hop addresses.
- Command **set ip default next-hop** is supported on GX, GX2, and FX3 platform switches.
- Use the optional **verify-availability** keyword to verify the reachability of the tracked object.

Verify 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.
show route-map [*name*] **pbr-statistics**	Displays policy statistics.

Use the **route-map** *map-name* **pbr-statistics** command to enable policy statistics. Use the **clear route-map** *map-name* **pbr-statistics** command to clear these policy statistics.

Configuration examples for policy-based routing

This topic provides configuration examples for policy-based routing, including route-map configuration, verification, and next-hop scenarios.

This example shows how to configure a simple route policy on an interface:

```
feature pbr
        ip access-list pbr-sample_1
        permit tcp host 10.1.1.1 host 192.168.2.1 eq 80
        ip access-list pbr-sample_2
        permit tcp host 10.1.1.2 host 192.168.2.2 eq 80
        !
        route-map pbr-sample permit 10
        match ip address pbr-sample_1
        set ip next-hop 192.168.1.1
        route-map pbr-sample permit 20
        match ip address pbr-sample_2
        set ip next-hop 192.168.1.2
        !
        route-map pbr-sample pbr-statistics
        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_1
        Set clauses:
        ip next-hop 192.168.1.1
        route-map pbr-sample, permit, sequence 20
        Match clauses:
        ip address (access-lists): pbr-sample_2
        Set clauses:
        ip next-hop 192.168.1.2
        switch#
        show route-map pbr-sample pbr-statistics
        route-map pbr-sample, permit, sequence 10
        Policy routing matches: 84 packets
        route-map pbr-sample, permit, sequence 20
        Policy routing matches: 94 packets
```

```
Default routing: 233 packets
```

Note **Policy routing matches** shown against every route-map sequence contains the number of packets in the incoming data traffic that has a match with the sequence in the route-map. This counter increments irrespective of whether the PBR redirection ('set' command of that sequence) is resolved or not. Correspondingly, in the example shown above, policy routing matches is shown against two route-map sequence (sequence 10 and 20) in the show route-map pbr-statistics pbr-sample output.

Note **Default routing** contains the number of packets in the incoming data traffic that has no match with any of the sequence in the route-map. Correspondingly, in the example shown above, default routing is shown only once at the end in the show route-map pbr-statistics pbr-sample output.

This example shows load sharing between ECMP and non ECMP paths:

```
switch#
show run rpm
!Command: show running-config rpm
!Running configuration last done at: Sun Dec 23 16:02:32 2018
!Time: Sun Dec 23 16:06:13 2018
version 9.2(3) Bios:version 08.35
feature pbr
route-map policy1 pbr-statistics
route-map policy1 permit 10
match ip address acl2
set ip next-hop 131.1.1.2 load-share
route-map policy2 pbr-statistics
route-map policy2 permit 10
match ip address acl2
set ip next-hop verify-availability 131.1.1.2 track 1
set ip next-hop verify-availability 30.1.1.2 track 2 load-share
interface Ethernet1/31
ip policy route-map policy2
```

This example displays information about next hop routing request:

```
switch#
show system internal rpm pbr ip nexthop
PBR IPv4 nexthop table for vrf default
30.1.1.2 Usable
via 28.1.1.2 Ethernet1/18 a46c.2ae3.02a7
131.1.1.2 Usable
via 111.1.1.2 Vlan81 8478.ac58.afc1
Usable
via 112.1.1.2 Vlan82 8478.ac58.afc1
Usable
via 113.1.1.2 Vlan83 8478.ac58.afc1
Usable
via 114.1.1.2 Vlan84 8478.ac58.afc1
Usable
via 115.1.1.2 Vlan85 8478.ac58.afc1
Usable
via 116.1.1.2 Vlan86 8478.ac58.afc1
```

```
        Usable
        via 117.1.1.2 Vlan87 8478.ac58.afc1
        Usable
        via 118.1.1.2 Vlan88 8478.ac58.afc1
```

This example display routes from the unicast RIB:

```
        switch#
        show ip route 130.1.1.2
        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>
        130.1.1.0/24, ubest/mbest: 8/0
        *via 111.1.1.2, Vlan81, [110/120], 00:07:57, ospf-1, inter
        *via 112.1.1.2, Vlan82, [110/120], 00:07:57, ospf-1, inter
        *via 113.1.1.2, Vlan83, [110/120], 00:07:57, ospf-1, inter
        *via 114.1.1.2, Vlan84, [110/120], 00:07:57, ospf-1, inter
        *via 115.1.1.2, Vlan85, [110/120], 00:07:57, ospf-1, inter
        *via 116.1.1.2, Vlan86, [110/120], 00:07:57, ospf-1, inter
        *via 117.1.1.2, Vlan87, [110/120], 00:07:57, ospf-1, inter
        *via 118.1.1.2, Vlan88, [110/120], 00:07:57, ospf-1, inter


        switch#
        show ip route 30.1.1.2
        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>
        30.1.1.0/24, ubest/mbest: 1/0
        *via 28.1.1.2, [1/0], 00:38:36, static
```

This example displays Policy-Based Routing with vrf-based next hop:

```
route-map policy_vrf_default_v4 permit 10
        match ip address acl1_v4_tc1
        set ip vrf default next-hop 31.1.1.1

route-map policy_vrf_nondefault_v4 permit 10
        match ip address acl1_v4_tc2
        set ip vrf vrf1 next-hop 32.1.1.1

show route-map policy_vrf_default_v4
        route-map policy_vrf_default_v4, permit, sequence 10
        Match clauses:
        ip address (access-lists): acl1_v4_tc1
        Set clauses:
        ip vrf default next-hop 31.1.1.1

show route-map policy_vrf_nondefault_v4
        route-map policy_vrf_nondefault_v4, permit, sequence 10
        Match clauses:
        ip address (access-lists): acl1_v4_tc2
        Set clauses:
        ip vrf vrf1 next-hop 32.1.1.1
```

This example displays Policy-Based Routing with default next hop:

```
route-map policy_default_v4 permit 10
        match ip address acl1_v4_tc1
        set ip default next-hop 21.1.1.2

show route-map policy_default_v4
        route-map policy_default_v4, permit, sequence 10
        Match clauses:
        ip address (access-lists): acl1_v4_tc1
        Set clauses:
        ip default next-hop 21.1.1.2
```

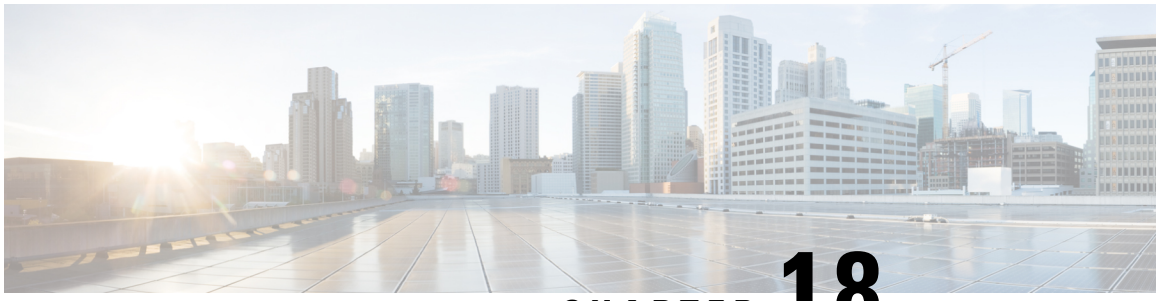
This example displays Policy-Based Routing with vrf-based default next hop:

```
route-map policy_default_vrf_default_v4 permit 10
          match ip address acl1_v4_tc1
          set ip default vrf default next-hop 21.1.1.2
          route-map policy_default_vrf_nondefault_v4 permit 10
          match ip address acl1_v4_tc1
          set ip default vrf vrf1 next-hop 22.1.1.2
          show route-map policy_default_vrf_default_v4
          route-map policy_default_vrf_default_v4, permit, sequence 10
          Match clauses:
          ip address (access-lists): acl1_v4_tc1
          Set clauses:
          ip default vrf default next-hop 21.1.1.2
          show route-map policy_default_vrf_nondefault_v4
          route-map policy_default_vrf_nondefault_v4, permit, sequence 10
          Match clauses:
          ip address (access-lists): acl1_v4_tc1
          Set clauses:
          ip default vrf vrf1 next-hop 22.1.1.2
```

Related documents for policy-based routing

This topic provides links to related documents for policy-based routing.

Related Topic	Document Title
IP SLA PBR object tracking	Cisco Nexus 9000 Series NX-OS IP SLAs Configuration Guide
Troubleshooting information	Cisco Nexus 9000 Series NX-OS Troubleshooting Guide



CHAPTER 18

Configure Dynamic Load Balancing

- Dynamic Load Balancing mechanism, on page 581
- Dynamic Load Balancing on CloudScale switches, on page 584
- Dynamic Load Balancing on Silicon One switches, on page 593
- Dynamic Load Balancing on NVIDIA Spectrum switches, on page 603
- Configuration samples for Dynamic Load Balancing, on page 605
- Verify the Dynamic Load Balancing configuration, on page 606
- Troubleshoot Dynamic Load Balancing, on page 608

Dynamic Load Balancing mechanism

Dynamic Load Balancing (DLB) is an advanced and intelligent hashing mechanism that

- enhances traditional ECMP forwarding,
- optimizes traffic distribution by considering link load, and
- dynamically directs traffic over underutilized links.

This occurs at the IP layer (Layer 3 in the OSI model) and is often implemented in modern networking hardware such as Nexus 9000 series switches.

ECMP is used to increase the bandwidth available to applications by allowing multiple parallel paths for traffic to flow between any two points in a network. When a router must forward a packet to a destination for which it has multiple equal-cost paths, it uses a hashing algorithm to decide which path to use for that packet. The algorithm typically takes into consideration parameters such as the source and destination IP addresses, source and destination port numbers, and sometimes even the protocol type.

In traditional load balancing, the path chosen for a given IP flow does not change over time unless there is a change in the network topology or manual reconfiguration by a network administrator. In contrast, Layer 3 ECMP Dynamic Load Balancing implies that the selection of the path can change according to the current state of the network. The router or switch can monitor the traffic load on each path and select a path with least link utilization to better distribute the traffic across all available paths. Thus, the Layer 3 ECMP DLB feature on the supported Nexus 9000 switches allows for the efficient distribution of traffic across multiple equal-cost paths in the network.

The Layer 3 ECMP DLB is supported along with RDMA over Ethernet (RoCE) with leaf-and-spine architecture that are used in the back-end of Artificial Intelligence and Machine Learning (AI/ML) training networks. A

fabric with DLB, when combined with PFC, along with ECN, provides an optimal network behavior by way of better utilization, low latency, and loss-less fabric.

Features

A few significant features of Dynamic Load Balancing include

- avoids the traditional hash-entropy problems with static ECMP load balancing,
- maximizes the utilization of available network paths,
- minimizes congestion by evenly spreading traffic across all paths,
- increases overall network performance without needing additional or specialized infrastructure, and
- provides faster convergence and redundancy in case of link or node failures

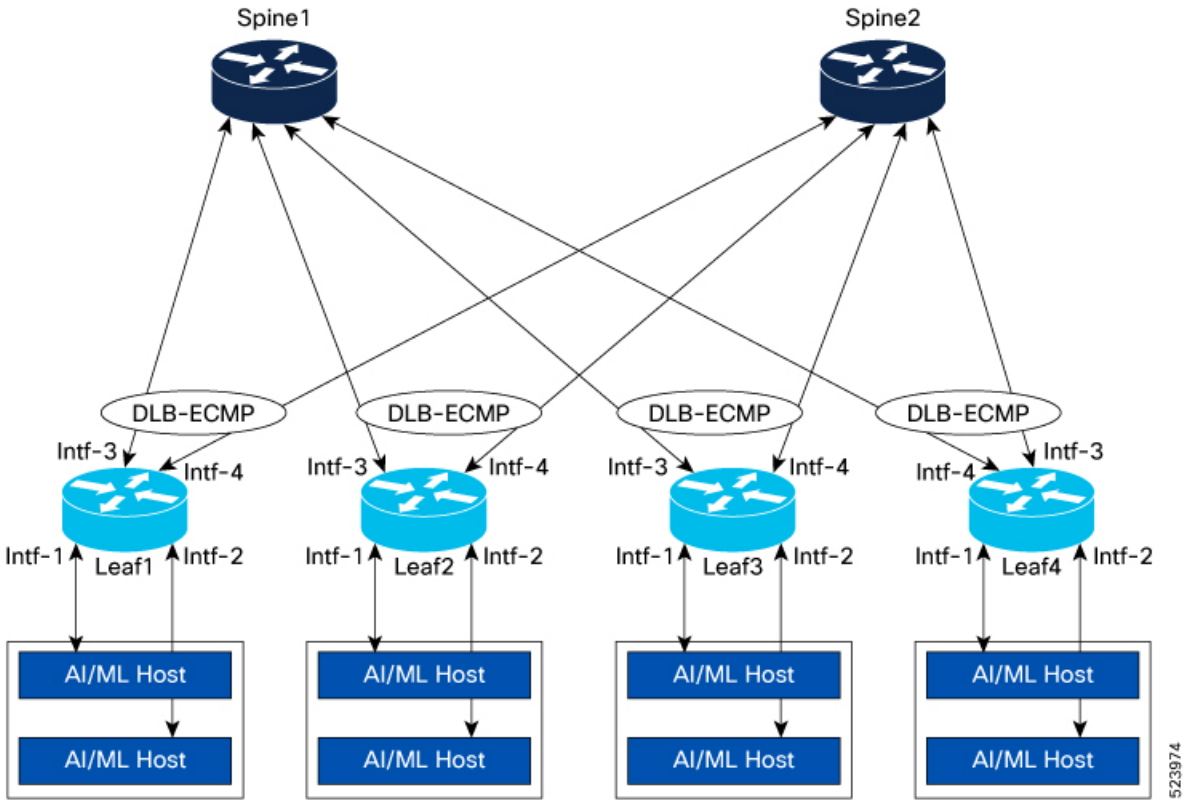
How DLB topology works in AI/ML networks

The Dynamic Load Balancing (DLB) topology is useful in AI/ML training networks. These networks use the spine-leaf architecture as shown in the image.

In this topology, the AI and ML hosts (server) are connected to Interface-1 (Intf-1) and Interface-2 (Intf-2) of the leaf switches. Intf-3 and Intf-4 of the leaf switches are connected to two spines, Spine 1 and Spine 2. While synchronizing data, for example, training data, across the AI/ML hosts, the training data gets transferred through the spine-leaf fabric among all the hosts.

Workflow

Figure 42: DLB Topology



Result

As leaf switches are connected to spines with more than one link, ECMP is used to load-share traffic across multiple links. The AI/ML training networks have fewer traffic flows with unique 5-Tuple IP fields compared to traditional networks. As such flows are limited in number, the traditional ECMP can lead to polarization issues, meaning suboptimal use of redundant paths, which in turn can lead to over-subscription on some links or interfaces. This can result in overall low throughput in the fabric.

The ECMP DLB feature resolves link utilization issues such as no utilization or under utilization by ensuring proper usage of all links. When you enable DLB on all the ports that are part of an ECMP group, a link with the lowest Tx link utilization is chosen among the available links for every new flow. In the image, DLB is enabled on Intf-3 and Intf-4. If intf-3 is fully utilized, and a new flow arrives, intf-4 gets selected. In traditional ECMP, the possibility is that Intf-3 gets picked even though it is oversubscribed.

ECMP DLB also supports static pinning, which allows users to pin traffic coming from a particular source port to be always sent on a specific DLB enabled egress port. In the image, for traffic taking a DLB ECMP group in which Intf-3 and Intf-4 are members, the user can pin traffic from Intf-1 to always take Intf-3 and Intf-2 to always take Intf-4.

Dynamic Load Balancing feature in NX-OS releases

Starting from Cisco NX-OS Release 10.5(1)F, the Layer 3 ECMP Dynamic Load Balancing (DLB) feature provides support to efficiently load balance traffic, depending on the current state of utilization of the outgoing links. The support for this feature is provided on different Nexus switches through different NX-OS releases as shown in the table.

- Beginning with Cisco NX-OS Release 10.5(1)F, the Layer 3 ECMP Dynamic Load Balancing (DLB) feature is supported on Nexus CloudScale switches such as 9300-FX3, -GX, -GX2, -H1, and -H2R TORs.
- Beginning with Cisco NX-OS Release 10.5(3)F, the Layer 3 ECMP Dynamic Load Balancing (DLB) feature is supported on Silicon One switches—only on N93C64E-SG2-Q and N9364E-SG2-O.
- Beginning with Cisco NX-OS Release 10.6(2n), the Layer 3 ECMP Dynamic Load Balancing (DLB) feature is supported on the Cisco N9164E-NS4-O switch.

Dynamic Load Balancing on Nexus switches

The concept and configuration of DLB feature differs on the CloudScale and Silicon One switches. For more information, see the two sections:

- Dynamic Load Balancing on CloudScale switches
- Dynamic Load Balancing on Silicon One switches

Dynamic Load Balancing on CloudScale switches

The concepts, guidelines, limitations, and configuration sections for CloudScale switches differ from the Silicon One switches and are described in this section.

Key concepts of Dynamic Load Balancing on CloudScale switches

This section comprises the main concepts that you need to know before configuring DLB on CloudScale switches.

Fast link failover

Fast link failover in the context of DLB ECMP load balancing on Nexus 9000 switches is a feature that allows the network to quickly respond to and recover from physical link failures. When a link used in an ECMP group fails, fast link failover ensures that the traffic is immediately redirected to the remaining operational links the link failover is detected by the hardware, and new link is selected from the remaining links automatically. As this occurs at the hardware layer, this provides faster convergence.

Dynamic Rate Estimator

A Dynamic Rate Estimator (DRE) is implemented in the hardware for measuring the current link utilization. The role of the DRE within DLB is to provide real-time estimation of the traffic rate on various links. This real-time analysis allows the switch to make more informed decisions when distributing traffic to ensure that no single path becomes over saturated. When a new flow starts, the DLB uses the DRE metric to determine the least utilized path within the multiple paths in a DLB ECMP Group.

At any point in time, a DLB-enabled interface can be at one of the DRE levels from level 1 to level 7 based on the utilization of the link and configured DRE thresholds. Level-1 indicates lowest utilization and level-7 the highest utilization. During the DLB decision, always a link with lowest DRE level is selected. If more than one link is available with the same lowest DRE level, one of the links among them is selected randomly. For more information, see Calculate Link Utilization Level.

Modes

Layer 3 ECMP Dynamic Load Balancing supports any one of the following modes in global configuration:

- Flowlet Load Balancing (FLB) – In this mode, load balancing occurs at flowlet level based on DRE metrics. This is the default mode.
- Per-packet Load Balancing (PLB) – In this mode, the load balancing decision occurs at a per-packet level instead of the flowlet level.

Flowlet Load Balancing

Flowlets are bursts of packets from a flow, identified by their 5-tuple (or selected fields from the packet), that are separated by large enough gaps such that they can be routed independently without causing reordering.

The flowlet is a unit of traffic used when the DLB works in flowlet mode. For each flowlet, the best outgoing port is picked by the hardware, which has the least Tx utilization, indicated by a per-port DRE. If utilization is the same for all ports, one of the ports is randomly selected.

Once a port is selected for a flowlet, the same port is used for all subsequent packets from that flow. A new port selection is triggered only when there is an inter-packet gap in the flowlet that is greater than the configured flowlet-aging time or when the currently utilized port goes down.

Per-packet Load Balancing

Per-packet Load Balancing can be used for scenarios where the end points (for example, Smart NICs) allow for packet re-ordering. This mode distributes traffic across the available links in a DLB ECMP and helps spread traffic out, reducing network congestion. For each packet in a flow, a new output port selection happens. So, the packets from the same flow can be sent across multiple paths causing packet re-ordering. The port selection process uses DRE, that is, port with the least DRE metric is selected for every packet. If DRE is the same for all ports, one of the ports is randomly selected.

Adaptive routing

When a Nexus switch peers with NVIDIA® NIC that supports adaptive routing, and you enable **hardware profile spectrum-x** on the NX-OS switch, the adaptive routing feature is enabled on the NIC.

The spectrum-x command enables NVIDIA's adaptive routing capabilities which include dynamically adjusting network routing paths based on traffic conditions, congestion, or changes in network topology. Handling out-of-order packets is one of the key capabilities of the NIC that are enabled in this mode. Combining the NX-OS switch's Per-packet Load Balancing (PLB) with this provides improved performance.

In a fabric requiring NVIDIA Spectrum-X adaptive routing, we recommend enabling **hardware profile spectrum-x** on all switches in the fabric that support this feature (both leaf and spine nodes) to ensure consistent telemetry and proper license compliance.

Static pinning

Static pinning is supported on DLB. In static pinning, a source port is pinned to a destination port that is part of a DLB enabled ECMP group. All the traffic from this source port is sent to the pinned destination port if

this port is part of the DLB ECMP group used for this flow. Front panel ports (including breakout ports) can be used as static pinning source interface. Destination interface must be part of the DLB interfaces list.

When static pinning is enabled, static pinning overrides the DLB DRE-based port selection.

When a DLB port is used as destination port in static pinning, this port cannot be removed from the dlb-interface list unless the static pinning configuration for that port is removed.



Note

- You can enable either static pinning or PLB mode. You cannot enable both at the same time. However, beginning with Release 10.6(2)F, you can enable static pinning in PLB mode. Static pinning takes precedence over PLB.
- Beginning with Cisco NX-OS Release 10.5(2)F, the system supports up to 512 static pinning pairs.

Guidelines and Limitations for Dynamic Load Balancing on CloudScale switches

The guidelines and limitations for Layer 3 Dynamic Load Balancing on CloudScale switches are categorized into four subsections that include

- ECMP group,
- Feature support,
- Ports, and
- DLB parameters.

ECMP group

- The decision to enable DLB should be done during ECMP group creation when these three conditions are met:
  - All members of an ECMP group are in the DLB-enabled interface list. If an ECMP group has one or more members that are not in the DLB interface list, regular ECMP is used for that ECMP group.
  - The members of an ECMP group can only be Layer 3 interfaces. Breakout ports, subinterfaces, SVI, and port-channels cannot be members of a DLB ECMP group.
  - ECMP is not weighted ECMP or Resilient ECMP.
- Resilient ECMP and DLB features cannot be enabled together. For more information about resilient ECMP, refer to Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide.
- For weighted ECMP groups, DLB is not applicable. For more information about weighted ECMP or UCMP, see Unequal Cost Multipath (UCMP) over BGP, on page 384.
- From Cisco NX-OS Release 10.6(3), DLB is supported for weighted ECMP groups.
- The show routing hash command does not work for routes using DLB ECMP Groups. This is because, when DLB is enabled, the port selection is done dynamically based on the link utilization instead of using static hash.

- In cases where the DLB ECMP scale is reached or if, due to any condition, DLB cannot be enabled, then the regular ECMP without DLB is used.

Note When one of the member ports in a DLB-enabled ECMP group goes down, the hardware immediately stops sending traffic through that port. This ensures minimal traffic loss during link failures.

Feature support

- This feature is supported only on
  - Layer 3 physical interfaces,
  - IP routed fabrics and VXLAN fabrics, and
  - 9300-FX3, GX, GX2, H1, and H2R TOR platforms.
- This feature is not supported on TORs that have line-card expansion modules (LEM) and N9K-C9408.
- Egress Access-list policies, Egress QoS policies, and TX SPAN configured on the egress interfaces are not applied for flows using ECMP DLB.
- MPLS/GRE tunnel do not use DLB ECMP, they fall back to regular ECMP.
- When you use this feature, if the DLB ECMP scale is high, we recommend using the system pic-core option. After using the system pic-core option, a switch reload is required. For more information, see Configuring BGP PIC Core.
- DLB is not applicable to traffic flows to which policy-based routing (PBR) logic is applied. These flows use the regular ECMP feature.
- MTU should be configured on all DLB-enabled interfaces based on the maximum size of the packets used in the DLB flows. Otherwise, as output drops, traffic is also dropped on egress interfaces.
- Only IPv4 and IPv6 unicast traffic is supported.
- Starting from Cisco NX-OS Release 10.6(1)F, as a part of the Al/ML License Enhancements, adaptive routing feature is supported on CloudScale switches and requires relevant licenses. Furthermore, DLB feature requires Advantage license from 10.6(1)F. When DLB is enabled and has a valid interface list, it consumes a license. For more information about licenses, refer to Cisco NX-OS Licensing Options Guide.
- Beginning with NX-OS Release 10.6(1)F, you can enable **hardware profile spectrum-x** on N9K-9364D-GX2A and N9K-9332D-GX2B switches in a fabric.

Ports

- Breakout ports, port-channels, Switch Virtual Interfaces (SVIs), port-channel members, or subinterfaces cannot be a part of the DLB-enabled interface list.
- A maximum of 63 physical ports can be part of the DLB interface list.

DLB parameters

This section lists the guidelines for DLB-related parameters such as MAC, aging, DRE thresholds, mode, and static pinning.

- A switch reload is required when a DLB interface list is configured for the first time or modified for the configuration to take effect.
- Choose the flowlet-aging time based on the round-trip time in the fabric; otherwise, the flows can be re-ordered.
- All the DLB-related parameters under DLB configuration are programmed in hardware only when there is a valid applied DLB interface list.
- When any one of the MAC, aging, mode, or DRE threshold configurations are removed, all parameters are set to default values.
- After adding a port to the DLB interface list, if the port is either modified to be a breakout port or added to a PO or if a subinterface is created on this interface, DLB is no longer enabled for ECMP groups that contain the port. Remove the port from the DLB interface list.
- Any change to the DRE thresholds can have a momentary traffic impact on DLB-enabled flows. This is a disruptive trigger.
- The DLB MAC configuration should be the same on all the nodes in the fabric. If the DLB MAC configuration is changed on a switch without changing this in the connected nodes in the fabric receiving these flows, the traffic is dropped.
- Static pinning and Per-packet DLB modes cannot be supported at the same time until Release 10.6(1)F. However, from Release 10.6(2)F, static pinning can be enabled for per-packet mode.
- Breakout ports can be a part of the static pinning source interfaces. However, subinterfaces and port-channels cannot be part of the static pinning source interfaces.
- Beginning with Release 10.6(2)F, policy-driven DLB mode parameter is supported.

Configure Dynamic Load Balancing on CloudScale switches

To configure Layer 3 Dynamic Load Balancing on CloudScale switches, run the commands listed in this section in the **hardware profile dlb** sub mode.

Before you begin

Use the **configure terminal** command to ensure that you are in the global configuration mode.

Procedure

Step 1 Use the **hardware profile dlb** command to enter the hardware profile dynamic load balancing mode. This command requires a reload.

Example:

```
switch(config)# hardware profile dlb
```

Step 2 Use the **dlb-interface** <*interface_range*> command to specify the list of interfaces for which DLB is to be enabled. Add comma-separated interfaces. This list cannot be changed dynamically, and requires switch reload for the interface list to be effective.



Note
- Any change in interface list requires reload for the configuration to be effective.
- To verify the current applied list, use the **show hardware profile dlb** command.
- Incremental addition or deletion to the interface-list is not supported. The configuration is replaced with the newly provided interface list.

Example:

```
switch(config-dlb)# dlb-interface Eth1/5,Eth1/7,Eth1/17,Eth1/21,Eth1/26
```

Step 3 (Optional) Enter the **dre-thresholds** [**level-1** *percentage_1* | **level-2** *percentage_2* | **level-3** *percentage_3* | **level-4** *percentage_4* | **level-5** *percentage_5* | **level-6** *percentage_6* | **level-7** *percentage_7*] command in DLB mode to define DRE levels from level 1 to level 7. The value configured per level is the percentage utilization range of port bandwidth from the previous level. The total of all the levels specified must be equal to 100. If you do not configure the DRE threshold levels, the following default values are used: 30, 20, 15, 10, 10, 10, and 5.

For more information about DRE level link utilization, see Calculate Link Utilization Level.

Example:

```
switch(config-dlb)# dre-thresholds level-1 15 level-2 20 level-3 30 level-4 15 level-5 10 level-6 5
 level-7 5
```

Step 4 (Optional) Use the **flowlet-aging** *usec* command in DLB mode to configure flowlet aging time in usecs. The default is 500 usecs and the maximum value is 2 seconds or 2000000 usecs.



Note Ensure that you choose the flowlet aging time with utmost care, else the flows can get re-ordered.

Example:

```
switch(config-dlb)# flowlet-aging 600
```

Step 5 (Optional) Use the **mac-address** *macaddr* command in DLB mode to configure the DLB MAC address. This address is used as next-hop MAC address for all the flows that use an ECMP Group created on DLB enabled interfaces . This DLB MAC is used to re-write the Destination MAC for DLB flows instead of the learned next-hop MAC address for the egress interfaces.

The guidelines and limitations include:

- If DLB is not enabled for flows using policy DLB, as long as the ECMP is on DLB enabled interfaces the MAC will be DLB MAC.
- If you do not configure this command, the default DLB MAC address used during the feature initialization is used as the default DMAC, that is, 00:CC:CC:CC:CC:CC.
- If you configure the DLB MAC address, then the default MAC is replaced with the newly configurable MAC address.
- All packets received on the switch with this DLB MAC as destination MAC are treated as routed packets.

- When applying this configuration, ensure that all other nodes in the fabric are configured with the same DLB MAC.
- If there is no dlb-interface list applied, then the DLB MAC cannot be used as additional router MAC.
- Broadcast and multicast MAC addresses cannot be configured as DLB MAC address.
- When CloudScale leaf switches connect to uplink spine switches, the spine must recognize the leaf's DLB MAC as a routed destination on every leaf-facing interface. If the spine is not a CloudScale DLB-enabled device, configure the leaf's DLB MAC as a secondary router MAC on the spine. Use the same DLB MAC configured on the leaf.

```
interface e1/2/1-2
  description On the spine to the leafs
  ip address x.x.x.x/31
  mac-address 00:cc:cc:cc:cc:cc
```

If the spine is a CloudScale device with DLB enabled, the default DLB MAC address, 00:cc:cc:cc:cc:cc, is automatically installed as a secondary router MAC. If a custom DLB MAC is configured, configure the same custom MAC on all fabric nodes.

Example:

```
switch(config-dlb)# mac-address aa:bb:cc:dd:ee:ff
```

Step 6 (Optional) Enter the **mode** [**flowlet** | **per-packet**| [**policy-driven**[**flowlet** | **per-packet**]]] command in DLB mode to enable either flowlet or per-packet DLB or policy-driven DLB mode. The default mode is **flowlet**.

Note
- For per-packet mode, static pinning cannot be enabled. Beginning with Release 10.6(2)F, static pinning can be enabled for per-packet mode.
- To change between **flowlet** and **per-packet** DLB modes, switch reload is not required.
- To change between **policy-driven flowlet** and **policy-driven per-packet** DLB modes, switch reload is not required.
- To change between DLB mode [**flowlet** | **per-packet**] and policy-driven DLB mode [**policy-driven** [**flowlet** | **per-packet**], reconfigure the DLB mode, and then reload the switch.

Example:

```
switch(config-dlb)# mode flowlet
```

Step 7 (Optional) Use the **static-pinning** command in DLB mode to configure the static pinning feature.

Note For static pinning, per-packet mode cannot be enabled. Beginning with Release 10.6(2)F, static pinning can be enabled for per-packet mode.

Example:

```
switch(config-dlb)# static-pinning
```

Step 8 (Optional) Use the **source** *source physical interface* **destination** *destination physical interface* command to configure source and destination interfaces for static pinning. However, these should be physical interfaces only, that is, the front panel Ethernet interfaces. SVI, port-channels, or sub-interface cannot be source or destination interface.

The guidelines and limitations include:

- The destination interface should be a part of the DLB applied or configured interface-list, but a source interface cannot be a part of this list.
- Beginning with Cisco NX-OS Release 10.5(2)F, interface in the DLB applied or configured interface-list can also be used as static pinning source interface.
- If the same source interface is used for two configurations, then the first destination interface gets replaced with the second destination interface, as the source interface is the same.
- Break-out ports can be configured as source interface.
- When no breakout or break-out operations are performed on the ports, the user should update the DLB or static pinning configuration.
- An interface cannot be deleted from the DLB interface list if it is configured as a destination interface in static pinning. To delete it, first remove the static pinning configuration and then delete the interface from the DLB interface list.

Example:

```
switch(config-dlb-static-pinning)# source ethernet 1/1 destination ethernet 1/2
```

Step 9 (Optional) Use the **hardware profile spectrum-x** command to enable the NVIDIA spectrum-x adaptive routing feature. When you run this command, the DLB functionality is based on the configuration under **hardware profile dlb**.



Note Ensure that you enable **feature lldp**. For more information, refer to Cisco Nexus 9000 Series NX-OS System Management Configuration Guide. The spectrum-x command does not require a reload.

Example:

```
switch(config)# hardware profile spectrum-x
```

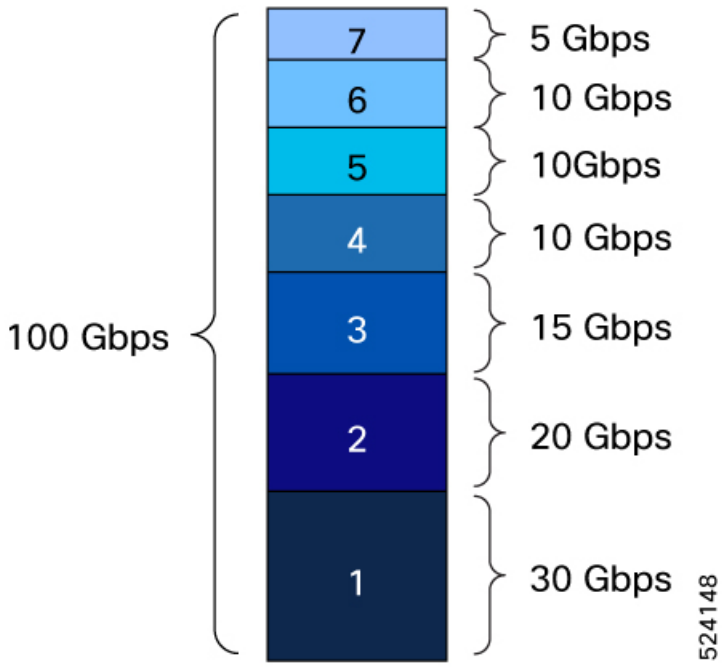
Calculate Link Utilization Level

The link utilization at each level is calculated as follows:

- Current Level Range Start: Sum of % values specified for all the previous levels.
- Current Level Range End: Current Level Range Start + Current Level value.

For example:

Level-1: 30, Level-2: 20, Level-3: 15, Level-4: 10, Level-5: 10, Level-6: 10, Level-7: 5.



- Level-5 Range Start: 75% (30 + 20 + 15 + 10)
- Level-5 Range End: 85% (75 + 10)

Guidelines for DRE Threshold Levels

A few guidelines and limitations related to Dynamic Rate Estimator (DRE) threshold levels are as follows:

- The above-mentioned logic applies to a scenario when any level is zero (0). You add up the levels, then that level will have the same range as the previous level with the non-zero value.

  For example,

  Level-1: 30, Level-2: 0, Level-3: 35, Level-4: 10, Level-5: 10, Level-6: 10, Level-7: 5.

  - Level-2 Range: Level 2 will be the same as Level-1, that is 30%.

- If all previous values are zero, the current level will be the first non-zero level specified.

  For example,

  Level-1: 0, Level-2: 0, Level-3: 0, Level-4: 0, Level-5: 50, Level-6: 30, Level-7: 20.

  - In this case, the starting level for link utilization will be Level-5.

- Specify all the levels. If a few levels are not specified, they will be considered as zero.

  For example,

  Level-1: 50, Level-2: 0, Level-3: 0, Level-4: 0, Level-5: 0, Level-6: 30, Level-7: 20

  - Levels 2, 3, 4, and 5 are considered zero.

Dynamic Load Balancing on Silicon One switches

The concepts, guidelines, limitations, and configuration sections for Silicon One switches differ from the CloudScale switches and are described in this section.

Key concepts of Dynamic Load Balancing on Silicon One switches

This section comprises the main concepts that you need to know before configuring DLB on Silicon One switches.

Flowlet Load Balancing

Flowlets are bursts of packets from a flow, identified by their 5-tuple (or selected fields from the packet), that are separated by large enough gaps such that they can be routed independently without causing reordering.

When DLB works in flowlet mode and a port is selected for a flowlet, the same port is used for all subsequent packets from that flow. This is called as Flowlet Load Balancing (FLB).

A new port selection is triggered only when there is an inter-packet gap in the flowlet that is greater than the configured flowlet-aging time or when the currently utilized port goes down. The new port selection happens at the start of each flowlet interval, ensuring uniform distribution of traffic across all links of the DLB ECMP group. The port selection process tries to uniformly distribute the packets across all the available ports at the flowlet boundary. The port selection can either be based on link utilization or be random.

Per-packet Load Balancing

Per-packet Load Balancing (PLB) can be used for scenarios where the endpoints, for example, Smart NICs, allow for packet re-ordering. This mode distributes traffic packet by packet across all the available links in a DLB ECMP group and helps spread out the traffic, reducing the network congestion. This mode offers the most optimal usage of the network bandwidth and faster end-to-end latency. For each packet in a flow, a new output port is selected. So, the packets from the same flow are sent across multiple paths potentially causing packet re-ordering.

The port selection process tries to uniformly distribute the packets across all the available ports. The port selection can either be based on link utilization or be random.

Policy-driven DLB

Policy-driven dynamic load balancing is a method used to manage the network traffic by dynamically classifying the ingress traffic flows based on predefined interface QOS policies and determining the load balancing mode for the flow.

Policies consist of rules that define when and how the dynamic load balancing should occur. These rules can be enabled or disabled, providing flexibility in managing the network traffic forwarding. When the traffic does not match the QOS policy rules with actions of flowlet or per-packet, policy override takes place and the traffic uses regular ECMP.

The policy- driven DLB mode relies on the QOS policy match, and the two types of policy-driven DLB we support include

- **Simple policy-driven DLB—**

Policy would only drive either flowlet or per-packet DLB behavior. The dlb mode in QoS Policy should match the mode in **hardware profile dlb** configuration.

- **Policy-driven DLB with mixed mode—**

  In this flexible mode of operation, system can support both flowlet and per-packet forwarding simultaneously. The ECMP scale in this profile is reduced from 512 ECMP groups to 256 ECMP groups.

Modes

Layer 3 ECMP Dynamic Load Balancing supports any one of the 7 modes in global configuration on Silicon One switches.

- **Flowlet Load Balancing (FLB)—**

  In this mode, load balancing occurs at flowlet level based on port load. This is the default mode when DLB is enabled.

- **Per-packet Load Balancing (PLB)—**

  In this mode, the load balancing decision occurs at a per-packet level instead of the flowlet level. This mode provides the most throughput with even utilization across all member ports but can result in packet reordering.

- **Policy-driven flowlet—**

  QoS **set dlb mode** option should be flowlet. When the mode is not matching the policy, applied interfaces go to err-disabled state.

- **Policy-driven per-packet—**

  QoS **set dlb mode** option should be per-packet. When the mode is not matching the policy, applied interfaces go to err-disabled state.

- **Policy-driven mixed (default ecmp)—**

  Without QoS Policy, we have regular ECMP based forwarding. We can have different QoS policies driving both flowlet and per-packet behavior.

- **Policy-driven mixed default flowlet—**

  By default the traffic takes flowlet DLB forwarding. A QoS policy can override the behavior to support per-packet forwarding.

- **Policy-driven mixed default per-packet—**

  By default the traffic takes per-packet DLB forwarding. A QoS policy can override the behavior to support flowlet-based DLB forwarding.

Choose and set the modes based on your traffic and scale requirements. In a default mode of operation, where you want DLB for all the traffic that is coming in on all the ports, use FLB or PLB. When you need DLB for selective traffic, for example, RDMA-over Converged Ethernet (RoCE)-based traffic and not for the regular traffic, use the mixed policy-driven DLB mode.

Adaptive routing

When a Nexus switch peers with NVIDIA® NIC that supports adaptive routing, and you enable **hardware profile spectrum-x** on the NX-OS switch, the adaptive routing feature is enabled on the NIC.

The spectrum-x command enables NVIDIA's adaptive routing capabilities which include dynamically adjusting network routing paths based on traffic conditions, congestion, or changes in network topology. Handling out-of-order packets is one of the key capabilities of the NIC that are enabled in this mode. Combining the NX-OS switch's Per-packet Load Balancing (PLB) with this provides improved performance.

In a fabric requiring NVIDIA Spectrum-X adaptive routing, we recommend enabling **hardware profile spectrum-x** on all switches in the fabric that support this feature (both leaf and spine nodes) to ensure consistent telemetry and proper license compliance.

Static pinning

Static pinning is supported on DLB. In static pinning, a source port is pinned to a destination port that is part of a DLB enabled ECMP group. All the traffic from this source port is sent to the pinned destination port if this port is part of the DLB ECMP group used for this flow. Front panel ports (including breakout ports) can be used as static pinning source interface. Destination interface must be part of the DLB interfaces list. The system supports up to 256 static pinning pairs.

When a DLB port is used as destination port in static pinning, this port cannot be removed from the dlb-interface list unless the static pinning configuration for that port is removed.

Fast link failover

Fast link failover in the context of DLB ECMP load balancing on Silicon One switches is a feature that allows the network to quickly respond to and recover from physical link failures. When a link used in an ECMP group fails, fast link failover ensures that the traffic is immediately redirected to the remaining operational links the link failover is detected by the hardware, and new link will be selected from the remaining links automatically. As this is done at the hardware layer, this provides faster convergence.

Hierarchical Weighted ECMP with DLB

In a spine-leaf topology, normally the traffic gets distributed equally across both the spines. However, when there are link failures in the fabric, this can result in congestion on one of the spines if not addressed at the ingress leaf. This problem can be resolved by considering link failures to account for the traffic that hits the spine on the ingress leaf using Weighted ECMP (WCMP).

The hierarchical WCMP provides the ability to pick the spines with full bandwidth towards a leaf into one DLB ECMP group and spines with partial bandwidth into another DLB ECMP group. The DLB is performed only among the members of the individual DLB ECMP group.

The hierarchical Weighted ECMP with DLB feature solves the problem while ensuring uniform distribution across all the links towards the spines after weighted distribution to maximize the link utilization.

DLB Static pinning only happens at level 2 of ECMP. If WCMP is enabled and all the weights are similar, static pinning works as usual because WCMP has only one member group at this level. If the weights are different, for example, because of link failures elsewhere in the network, static pinning still works only at level 2 ECMP. This means that only the traffic matching a specific weight is pinned to the port.

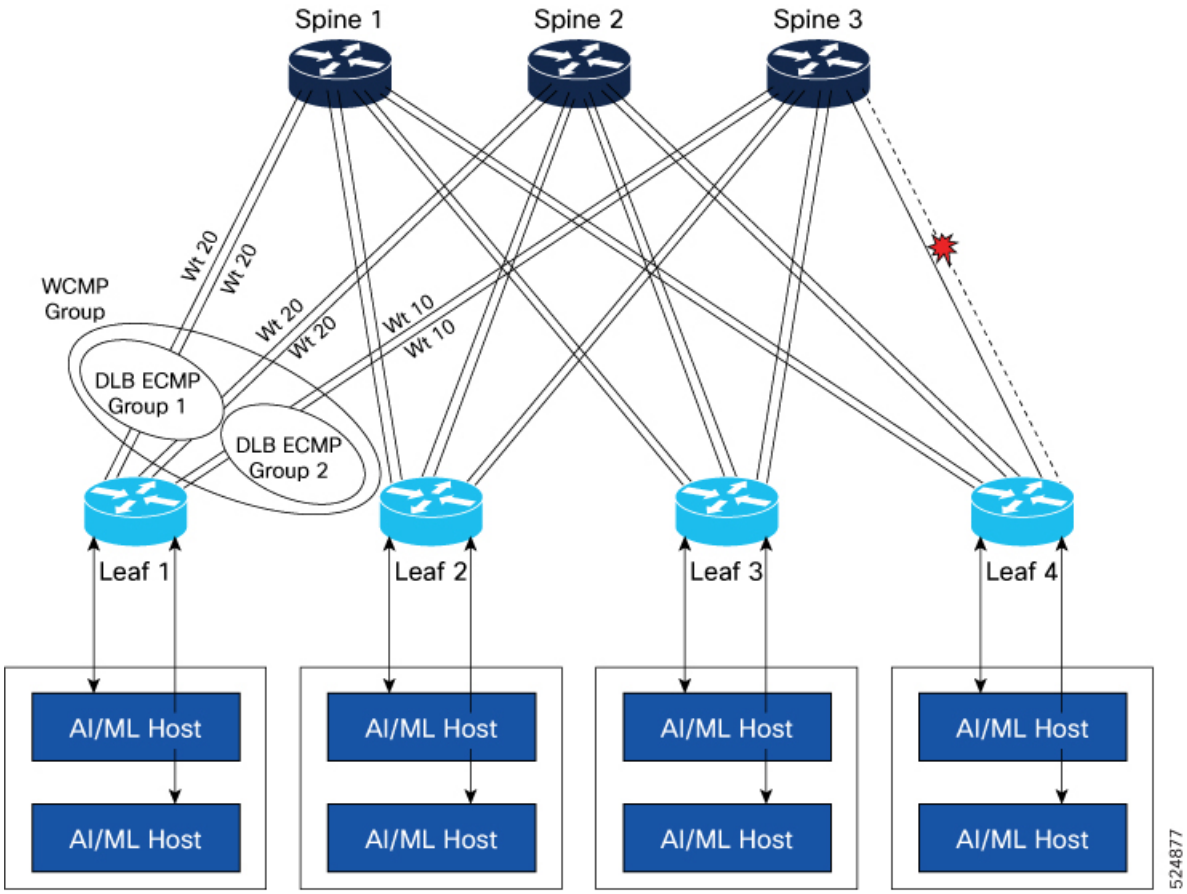
For more information regarding Weighted ECMP, also known as UCMP, refer to the Unequal Cost Multipath (UCMP) over BGP section in the Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide on Cisco.com.

An example for hierarchical Weighted ECMP is provided here for your reference.

The image depicts a topology where 3 spines connected to 4 leafs which are in turn connected to individual AI/ML hosts. The eBGP is configured on every link connecting Leafs to Spines. Leafs advertise routes with weights to the spine and the spines advertise cumulative weights.

Consider a 100-Gig traffic that is supposed to egress from Leaf 1 and ingress to Leaf 4 through Spines 1, 2, and 3. In the traditional way, it goes over a 6-way ECMP of all the links to Spine 1, 2, and 3. In a scenario where one link is down from Spine 3 to Leaf 4, the bandwidth available on that link is reduced compared to the bandwidth available on other Spines. This can result in congestion on Spine 3.

Figure 43: Hierarchical WCMP



This issue is resolved using hierarchical WCMP of DLB ECMP group. This provides the ability to pick Spines with full bandwidth towards a leaf into one group (DLB ECMP Group1) and Spines with partial bandwidth into another group (DLB ECMP Group2) and these together form the Weighted ECMP group and the FLB is performed among the members of this group. The WCMP dynamically picks Spines 1 and 2 for 80% traffic based on their bandwidth and Spine 3 for 20% of the traffic destined towards Leaf4. Thus, the traffic of weight 20 each is carried through the 2 links each of Spine 1 and 2 to Leaf4 while the traffic of weight 10 each is carried to 2 links of Spine 3 and delivered through one active link carrying traffic of weight 20 to Leaf 4.

In summary, the WCMP with DLB ECMP feature uses dynamic load balancing to select the best link based on link utilization.

IP load sharing

The flowlet is uniquely identified using the parameters configured in the **ip load-sharing** command. By default, the parameters used are source IP, destination IP, destination port, source port, and IPv4 protocol.

For more information about *Configuring ECMP Load Balancing* including Silicon One switches, refer to Interfaces Configuration Guide on Cisco.com.

Guidelines and limitations for Dynamic Load Balancing on Silicon One switches

The guidelines and limitations for Layer 3 Dynamic Load Balancing on Silicon One switches are divided into four sub-sections.

- Feature support,
- Ports,
- ECMP, and
- DLB parameters.

Feature support

- Layer 3 ECMP DLB is supported only on
  - Layer 3 physical interfaces,
  - IP routed fabrics and VXLAN fabrics, and
  - from 10.5(3)F, on N9364E-SG2 switches.
- If the DLB ECMP scale is high, use the system pic-core option and then reload the switch. For more information, see Configuring BGP PIC Core.
- Configure the MTU on all DLB-enabled interfaces based on the maximum packet size used in the DLB flows. If not, the traffic drops when the output drops on the egress interfaces.
- Only IPv4 and IPv6 unicast traffic is supported.
- Beginning with NX-OS Release 10.6(1)F, as a part of the Al/ML License Enhancements, adaptive routing feature is supported on Silicon One switches and requires relevant licenses. Furthermore, DLB feature requires Advantage license from 10.6(1)F. When DLB is enabled and has a valid interface list, it consumes a license. For more information about licenses, refer to Cisco NX-OS Licensing Options Guide.
- Beginning with NX-OS Release 10.6(1)F, you can enable **hardware profile spectrum-x** on N9364E-SG2-Q and N9364E-SG2-O switches in a fabric.
- Beginning with Release 10.6(2)F, static pinning is supported.

Ports

- Port-channels, SVIs, port-channel members, or sub-interfaces cannot be part of the DLB interface list.
- The DLB interface list can include up to 254 physical ports, including breakout ports from NX-OS Release 10.6(1)F. The **dlb interface all** option allows all physical ports in the system to support DLB.
- We support only one IPv4, one IPv6 nexthop, and one IPv6 link-local nexthop on a DLB interface.

ECMP

- The maximum support for members in a DLB ECMP is a 127-way ECMP.

- DLB policy-driven mixed mode reduces the supported scale from 512 to 256 groups.
- DLB is enabled on the ECMP group in the hardware during the group creation provided the listed conditions are met which include:
  - All members of an ECMP group are in the DLB-enabled interface list. If an ECMP group has one or more members that are not in the DLB interface list, regular ECMP is used for that ECMP group.
  - When you choose the **interface-list all** option, all Layer 3 physical interfaces including break-out interfaces on the switch can be the members of a DLB ECMP group.
  - Sub-interfaces, SVIs, and port-channels cannot be members of a DLB ECMP group. If any of these are members of an ECMP group, then DLB is not enabled for that group.
  - ECMP is not Resilient ECMP.
- Resilient ECMP and DLB features cannot be enabled together. For more information about resilient ECMP, refer to Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide.
- For the traditional weighted ECMP groups, DLB is not applicable. For more information about weighted ECMP or UCMP, see Unequal Cost Multipath (UCMP) over BGP, on page 384.
- The **show routing hash** command does not work for routes using DLB ECMP Groups. This is because, when DLB enabled, the port selection is done dynamically based on the link utilization instead of using static hash.
- For weighted ECMP to work on 400GB interface, you need the route-map configuration to set an bandwidth in spine and map it to BGP neighbor configuration towards leaf. This is because load balance value generated for BGP community is an high integer which is not recognized LB:65201:50000000000.

  Execute this configuration on leaf and spine on a 2-Clos topology and configure the route map, where the bandwidth is set, to the spine.

  Configuration for spine:

```
router bgp <>
  neighbor <>
    address-family ipv4 unicast
      send-community
      send-community extended
      route-map WCMP in
      dmz-link-bandwidth
      link-bandwidth cumulative
route-map WCMP permit 10
  set extcommunity bandwidth <>
```

  Configuration for leaf:

```
router bgp <>
  neighbor <>
    address-family ipv4 unicast
      send-community
      send-community extended
      link-bandwidth cumulative
```

Note When one of the member ports in a DLB-enabled ECMP group goes down, the hardware immediately stops sending traffic through that port. This ensures minimal traffic loss during link failures.

DLB parameters

The DLB parameters include the parameters such as aging and mode.

- When you configure a DLB interface list for the first time or modify it, reload the switch for the configuration to take effect.
- Choose the Flowlet-aging time based on the round-trip time in the fabric; otherwise, the flows can be re-ordered.
- Changing between DLB modes is allowed, but requires a switch reload. For DLB policy-based actions applied to the interfaces, ensure that the QoS configuration matches the new **dlb mode**.
- When you remove parameters such as aging, mode, decay-factor, sampling-interval, and load-awareness, they reset to their default values.
- After adding a port to the DLB interface list, if the port is either modified to be a breakout port or added to be a part of a PO or if a sub interface is created on this interface, DLB is no longer enabled for ECMP groups that contain the port. Remove the port from the DLB interface list.

Configure Dynamic Load Balancing on Silicon One switches

This procedure outlines the steps to configure Dynamic Load Balancing (DLB) on the Silicon One switch. Note that many configuration changes require a switch reload to take effect.

Note For the policy-driven DLB to work, perform the DLB configuration mentioned in this section and then configure the Quality of Service (QoS) policy with set dlb mode. See the Configure Policy-driven Dynamic Load Balancing section in the Cisco Nexus 9000 Series NX-OS Quality of Service Configuration Guide on Cisco.com.

SUMMARY STEPS

1. Enter the hardware profile dlb mode.
2. Specify interfaces for Dynamic Load Balancing (DLB). Use the **dlb-interface** [*interface_range* | **all**] command to specify the list of interfaces for which DLB is to be enabled. Add comma-separated interfaces. This list cannot be changed dynamically, and requires switch reload for the interface list to be effective.
3. (Optional) Configure flowlet aging time. Enter the **flowlet-aging** *usec* command in DLB mode to configure Flowlet aging time in microseconds (*usec*). The default value is 255 microseconds, and the maximum value is 1024 microseconds.
4. (Optional) Configure Dynamic Load Balancing mode.
5. (Optional) Adjust the load balancing behavior of DLB. Fine-tune how DLB responds to traffic changes and collects samples.
6. (Optional) Configure static pinning.
7. (Optional) Enable NVIDIA® spectrum-x adaptive routing. Use the **hardware profile spectrum-x** command to enable the NVIDIA spectrum-x adaptive routing feature. When you run this command, the DLB functionality is based on the configuration under **hardware profile dlb**.

DETAILED STEPS

Procedure

Step 1 Enter the hardware profile dlb mode.

a) Use the **configure terminal** command to ensure that you are in the global configuration mode.

Example:

```
switch# configure terminal
```

b) Use the **hardware profile dlb** command to enter the hardware profile dynamic load balancing mode. This command requires a reload.

Example:

```
switch(config)# hardware profile dlb
```

Note Run the commands listed in this section under the **hardware profile dlb** sub mode to configure Layer 3 Dynamic Load Balancing on Silicon One switches.

Step 2 Specify interfaces for Dynamic Load Balancing (DLB). Use the **dlb-interface** [*interface_range* | **all**] command to specify the list of interfaces for which DLB is to be enabled. Add comma-separated interfaces. This list cannot be changed dynamically, and requires switch reload for the interface list to be effective.

Note
- Any change in interface list requires reload for the configuration to be effective.
- Incremental addition or deletion to the interface-list is not supported. The configuration is replaced with the newly provided interface list.

When you use the **all** option, all Layer 3 physical and Layer 3 breakout interfaces on the switch should be enabled for DLB; this requires a switch reload to be effective,

Under the effective **dlb-interface all** mode, newly added DLB-capable interfaces do not require a reload.

Example:

```
switch(config-dlb)# dlb-interface Eth1/5,Eth1/7,Eth1/17,Eth1/21,Eth1/26
switch(config-dlb)# dlb-interface all
```

From Release 10.6(1)F, you can either perform one or more of the actions that include

- add breakout ports in the DLB interface list, for example, first 64 broken out 4-way ports,
- use the range command to add a range of interfaces, and
- use a combination of interfaces and range of interfaces.

Note

- You can add 64 interfaces or range of interfaces or a mix of both at the maximum.
- You can execute the **no interface breakout**... command to remove dlb-interface configurations and perform a system reload per the reload warning.

  If you break the ports again, re-apply the dlb-interface configurations. However, new configurations on the broken-out ports that are already present in the applied dlb-interface list are honored for FLB without a reload.

Example:

```
switch(config-dlb)# dlb-interface eth1/1/1-4, eth1/2/1-4, eth1/3/1-4 … eth1/64/1-2
switch(config-dlb)# dlb-interface eth1/1, eth1/2/1-4, eth1/3/1-4
switch(config-dlb)# no interface breakout module 1 port 1 map 100g-4x
```

a) Verify the interface list and pending reloads.

- To verify the currently applied list, use the **show hardware profile dlb** command.
- To know about pending reloads, use the **show system config reload-pending** command.

Step 3 (Optional) Configure flowlet aging time. Enter the **flowlet-aging** *usec* command in DLB mode to configure Flowlet aging time in microseconds (*usec*). The default value is 255 microseconds, and the maximum value is 1024 microseconds.

Note Choose the Flowlet aging time carefully to prevent flow reordering.

Example:

```
switch(config-dlb)# flowlet-aging 600
```

Step 4 (Optional) Configure Dynamic Load Balancing mode.

a) Enter the **mode** [**flowlet** | **per-packet** | **policy-driven**] command in DLB mode to enable flowlet, per-packet, or policy-driven DLB mode. The default mode is **flowlet**.

Note Reload the switch after changing the mode to apply the changes.

Example:

```
switch(config-dlb)# mode flowlet
```

b) Enter the **mode policy-driven** [**flowlet** | **per-packet** | **mixed**] command in DLB mode to enable QoS policies to drive flowlet, per-packet, or mixed policy-driven DLB mode. The mode set by the QoS policy should match the policy-driven DLB mode.

In mixed mode, both Flowlet and Per-packet are supported and the ECMP group scale is reduced from 512 to 256. Any one mode can be set as the default.

When the traffic does not match the QoS policy rules with actions of flowlet or per-packet, it defaults to using regular ECMP.

In the mixed mode, the default behavior can be modified using the configuration mentioned in Step 6.

For more information about the QoS policy match, refer to the Configure Policy-driven Dynamic Load Balancing chapter in the Cisco Nexus 9000 Series NX-OS Quality of Service Configuration Guide.

Note If you change the mode, reload is required.

Example:

```
switch(config-dlb)# mode policy-driven mixed default ecmp
```

c) Enter the **mode policy-driven mixed default** [**ecmp** | **flowlet** | **per-packet**] command in DLB mode to modify the default behavior in the mixed mode. When the traffic does not match the QoS policy rules with actions of flowlet or per-packet, it chooses the default mode configured using this command. If no mode is configured, the configuration defaults to regular ECMP.

Note If you change the mode, reload is required.

Example:

```
switch(config-dlb)# mode policy-driven mixed default ecmp
```

Step 5 (Optional) Adjust the load balancing behavior of DLB. Fine-tune how DLB responds to traffic changes and collects samples.

a) In DLB mode, use the **decay-factor** command to control how load balancing is influenced by the past traffic samples. Using a smaller value causes a faster response to changes as the data from the recent past is considered. A larger value results in a slower response to changes, as the emphasis is more on the historical data. The default value is 2. The decay-factor value ranges from 0 to 15.

Example:

```
switch(config-dlb)# decay-factor 2
```

b) Use the **sampling-interval** command in DLB mode to set the time interval for each port load sample collection in nanoseconds (nsecs). The default is 32000 nanoseconds. The range is 512 nanoseconds to 16,384,000 nanoseconds. The supported values are 512; 1000; 2000; 4000; 8000; 16,000; 32,000; 64,000; 128,000; 256,000; 512,000; 1,024,000; 2,048,000; 4,096,000; 8,192,000; and 16,384,000.

Example:

```
switch(config-dlb)# sampling-interval 32000 nsecs
```

c) Enter the [**no**] **load-awareness** command in DLB mode to use port load for port selection. The default value is **load-awareness** .

To ignore the port load, use the **no load-awareness** command.

Example:

```
switch(config-dlb)# load-awareness
```

Step 6 (Optional) Configure static pinning.

a) Use the **static-pinning** command in DLB mode to configure the static pinning feature.

Example:

```
switch(config-dlb)# static-pinning
```

b) Use the **source** *source physical interface* **destination** *destination physical interface* command to configure source and destination interfaces for static pinning. However, these should be physical interfaces only, that is, the front panel Ethernet interfaces. SVI, port-channels, or sub-interface cannot be source or destination interface.

Example:

```
switch(config-dlb-static-pinning)# source ethernet 1/9 destination ethernet 1/31
```

The guidelines and limitations for this command are:

- The destination interface should be a part of the DLB applied or configured interface-list.
- Interfaces in the DLB applied or configured interface-list can also be used as static pinning source interface.
- If the same source interface is used for two configurations, then the first destination interface gets replaced with the second destination interface, as the source interface is the same.
- Break-out ports can be configured as source interface, destination interface, or both.
- When no breakout or break-out operations are performed on the ports, the user should update the DLB or static pinning configuration.
- An interface cannot be deleted from the DLB interface list if it is configured as a destination interface in static pinning. To delete it, first remove the static pinning configuration and then delete the interface from the DLB interface list.
- When static pinning is enabled in policy-driven mode, policy is not be honored. Static pinning takes precedence over policy.
- When the destination interface goes down, traffic will be forwarded using the default DLB mode.

Step 7 (Optional) Enable NVIDIA® spectrum-x adaptive routing. Use the **hardware profile spectrum-x** command to enable the NVIDIA spectrum-x adaptive routing feature. When you run this command, the DLB functionality is based on the configuration under **hardware profile dlb**.

Note Ensure that you enable **feature lldp**. For more information, refer to Cisco Nexus 9000 Series NX-OS System Management Configuration Guide. The spectrum-x command does not require the switch to be reloaded.

Example:

```
switch(config)# hardware profile spectrum-x
```

Dynamic Load Balancing on NVIDIA Spectrum switches

Dynamic Load Balancing on NVIDIA Spectrum switches refers to the unique concepts, guidelines, limitations, and configuration procedures that differ from those for CloudScale and Silicon One switches.

- Concepts, guidelines, and limitations are specific to NVIDIA Spectrum switches.
- Configuration steps for Dynamic Load Balancing differ from other switch platforms.

Key concepts of Dynamic Load Balancing on NVIDIA Spectrum switches

Per-packet Load Balancing

Per-packet Load Balancing (PLB) is supported on NVIDIA Spectrum switches.

Per-packet Load Balancing (PLB) can be used for scenarios where the endpoints, for example, Smart NICs, allow for packet re-ordering. This mode distributes traffic packet by packet across all the available links in a DLB ECMP group and helps spread out the traffic, reducing the network congestion. This mode offers the most optimal usage of the network bandwidth and faster end-to-end latency. For each packet in a flow, a new output port is selected. So, the packets from the same flow are sent across multiple paths potentially causing packet re-ordering. The port selection process tries to uniformly distribute the packets across all the available ports. The port selection can either be based on link utilization or be random.

Adaptive routing

On NVIDIA Spectrum switches, adaptive routing is enabled by default.

Adaptive routing is a technique that dynamically selects the best network path for each packet or flow based on current network conditions, such as congestion or link failures. This helps optimize network performance, balance traffic loads, and improve overall reliability by automatically adjusting routing decisions in real time.

Weighted ECMP with DLB

Weighted ECMP with DLB is supported on NVIDIA Spectrum switches.

Weighted ECMP with DLB is supported on NVDIA Spectrum switches for IPv4 and IPv6 routes learned from BGP over global IP neighbors or link-local v6 neighbors.

With DLB enabled, packet spray over WCMP paths has following limitations:

- Weights are not honored when spraying traffic over WCMP from a leaf node to a single spine node.
- From a leaf node to multiple spines, packet spray to each spine works according to weight- provided all paths to a particular spine node have equal cost.

Unsupported DLB Features

Flowlet Load Balancing and Policy-driven DLB are not supported on NVIDIA Spectrum switches.

800G native speed is not supported on NVIDIA Spectrum switches.

Guidelines and limitations for Dynamic Load Balancing on NVIDIA Spectrum switches

DLB parameters

- On NVIDIA Spectrum switches, only per-packet DLB is supported. No other DLB parameters are configurable.
- The **show routing hash** command is not supported on NVIDIA Spectrum switches.

Configure Dynamic Load Balancing on NVIDIA Spectrum switches

Use this procedure to configure Layer 3 Dynamic Load Balancing (DLB) on NVIDIA Spectrum switches.

Procedure

Step 1 Enter the hardware profile DLB mode.

a) Use the **configure terminal** command to ensure that you are in the global configuration mode.

Example:

```
switch# configure terminal
```

b) Use the **hardware profile dlb** command to enter the hardware profile dynamic load balancing mode. This command requires a reload.

Example:

```
switch(config)# hardware profile dlb
```

Note Run the commands listed in this section under the hardware profile dlb sub mode to configure Layer 3 Dynamic Load Balancing on NVIDIA Spectrum switches.

Step 2 Verify the Dynamic Load Balancing configuration.

Example:

```
show hardware profile dlb
```

Displays the DLB configuration.

Configuration samples for Dynamic Load Balancing

Use the configuration samples as a reference to enable DLB on Nexus 9000 series switches. Separate configuration examples are provided for CloudScale and Silicon One hardware platforms.

Use this configuration for CloudScale switches.

```
switch# configure terminal switch(config)# hardware profile dlb
switch(config-dlb)# dlb-interface Eth1/5,Eth1/7,Eth1/17,Eth1/21,Eth1/26
switch(config-dlb)# dre-thresholds level-1 15 level-2 20 level-3 30 level-4 15 level-5 10
level-6 5 level-7 5
switch(config-dlb)# flowlet-aging 600
switch(config-dlb)# mac-address aa:bb:cc:dd:ee:ff
switch(config-dlb)# mode flowlet
switch(config-dlb)# static-pinning
switch(config-dlb-static-pinning)# source ethernet 1/1 destination ethernet 1/2
```

Use this configuration for Silicon One switches.

```
switch# configure terminal
switch(config)# hardware profile dlb
```

```
switch(config-dlb)# dlb-interface all
switch(config-dlb)# flowlet-aging 600
switch(config-dlb)# mode flowlet
switch(config-dlb)# decay-factor 2
switch(config-dlb)# sampling-interval 32000 nsecs
switch(config-dlb)# load-awareness
switch(config-dlb)# static-pinning
switch(config-dlb-static-pinning)# source ethernet 1/9 destination ethernet 1/31
```

Verify the Dynamic Load Balancing configuration

Choose one of the commands from the table to view information about the Dynamic Load Balancing (DLB) configuration. The table lists the show commands for DLB and their purposes.

Command	Purpose
show hardware profile dlb	Displays the DLB configuration. **Note** • Configured Interface-list—Lists current interfaces configured using the Command Line Interface. Post reload, this list populates the applied interface-list. • Applied Interface-list—Provides the list of interfaces that are being used currently for DLB. • Configured DLB mode—Provides the configured DLB mode. After reloading the switch this becomes the Applied DLB mode. • Applied DLB mode—Provides the DLB mode currently in use.
show hardware spectrum-x	Displays whether the spectrum-x adaptive routing feature is enabled.
show system config reload-pending	Displays the reload-pending configuration. For DLB, it displays the interface-list that is pending application if changes were made to the interface-list configuration.

Note The **show routing hash** command does not work for routes that use DLB ECMP Groups.

Show command output

You can view the sample output of the **show hardware profile dlb** command on CloudScale platforms.

```
switch# show hardware profile dlb
DLB Configurations:
===================
```

```
Enabled:                yes
Mode:                   flowlet
Mac-address:            aa:bb:cc:dd:ee:ff
Flowlet aging time:     600 usec(s)
DRE-thresholds:
        Level-1:15
        Level-2:20
        Level-3:30
        Level-4:15
        Level-5:10
        Level-6:5
        Level-7:5
DLB interface list:
--------------------

Configured Interface-list (size: 5):
        Eth1/5,Eth1/7,Eth1/17,Eth1/21,Eth1/26

Applied interface-list (size: 5):
        Eth1/5,Eth1/7,Eth1/17,Eth1/21,Eth1/26

Static-pinning enabled: yes

DLB static-pinning pairs:
---------------------------

static-pinning pairs (1):

        source: Eth1/1    dest: Eth1/5
```

You can view the sample output of the **show hardware profile dlb** command for NVDIA Spectrum switches:

```
switch# show hardware profile dlb
DLB Configurations:
===================


Enabled:                yes
Configured DLB Mode:    per-packet
Applied DLB Mode:       per-packet
DLB interface list:
--------------------

Configured Interface-list: all
Applied interface-list: all
```

You can view the sample output of the **show hardware profile dlb** command on Silicon One platforms.

```
switch# show hardware profile dlb
DLB Configurations:
===================


Enabled: yes
Configured DLB Mode: flowlet
Applied DLB Mode: flowlet
Flowlet aging time: 512 usec(s)
Decay Factor: 2
Sampling Interval: 32000 nsec(s)
Load Awareness: True
DLB interface list:
--------------------
Configured Interface-list : all
```

```
Applied Interface-list : all

Static-pinning enabled: yes

DLB static-pinning pairs:
--------------------------

static-pinning pairs (1):

source: Eth1/9 dest: Eth1/31

switch#
```

You can view the sample output of the **show hardware spectrum-x** command on all supported Nexus 9000 series switches, both CloudScale and Silicon One.

```
switch# show hardware profile spectrum-x
Spectrum-x Configurations:
==================
Enabled: yes
spine1(config)#
```

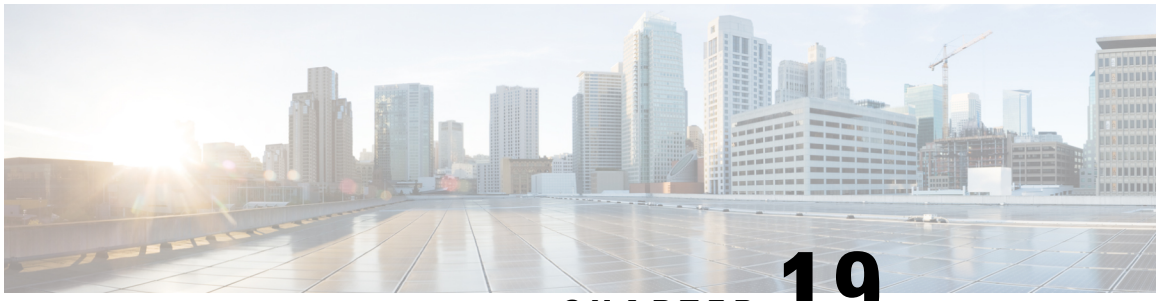
You can view the sample output of the **show system config reload-pending** command for DLB on the Nexus 9000 series switches, both CloudScale and Silicon One.

```
switch# show system config reload-pending

Following config commands require copy r s + reload :
=====================================================
0       hardware profile dlb ; dlb-interface Eth1/5,Eth1/7,Eth1/17,Eth1/21,Eth1/26
=====================================================
```

Troubleshoot Dynamic Load Balancing

For more information about troubleshooting DLB for all Nexus switches, refer to the Troubleshoot Dynamic Load Balancing section in Cisco Nexus 9000 Series NX-OS Troubleshooting Guide on Cisco.com.



CHAPTER 19

Configuring HSRP

This chapter contains the following sections:

- About HSRP, on page 609
- HSRP subnet VIP, on page 614
- HSRP authentication, on page 614
- HSRP messages, on page 614
- HSRP load sharing, on page 615
- Object tracking and HSRP, on page 616
- vPCs and HSRP, on page 617
- BFD, on page 618
- High availability and extended nonstop forwarding, on page 618
- Virtualization support, on page 619
- Prerequisites for HSRP, on page 619
- Guidelines and limitations for HSRP, on page 619
- Default settings for HSRP parameters, on page 621
- Configue HSRP, on page 622
- Verify the HSRP Configuration, on page 638
- Configuration examples for HSRP, on page 639
- Additional references, on page 640

About HSRP

HSRP is a protocol that enables a group of routers to provide first-hop routing redundancy by selecting an active router and a standby router, ensuring continuous packet routing even if the active router fails.

- Allows transparent failover of the first-hop IP router.
- Provides redundancy for IP hosts on Ethernet networks with a default router IP address.
- Uses a group of routers to select an active router (routes packets) and a standby router (takes over if the active fails).

Many host implementations do not support dynamic router discovery mechanisms but can be configured with a default router. HSRP provides failover services to these hosts, addressing administrative, processing, and security concerns associated with running dynamic router discovery on every host.

HSRP overview

HSRP (Hot Standby Router Protocol) is a redundancy protocol that allows a group of routers to share a *virtual IP address* and *virtual MAC address* , so that hosts can use a single default gateway for reliable network connectivity.

- Routers in an HSRP group share a *virtual IP address* and *virtual MAC address* as the default gateway for hosts.
- One router is elected as the *active router* to forward packets, while another is selected as the *standby router* to take over if the active router fails.
- HSRP uses priority values and hello messages to determine active and standby roles and to detect failures.

HSRP operates by configuring a virtual IP and MAC address on each participating router interface. The active router handles traffic for the virtual addresses, while the standby router monitors the active router and takes over if it fails.

- Each HSRP-enabled interface is configured with the same virtual IP and MAC address, and a unique real IP and MAC address.
- Routers send and receive multicast UDP-based hello messages to detect failures and manage role transitions.
- HSRP can support multiple groups on a single interface for additional redundancy.

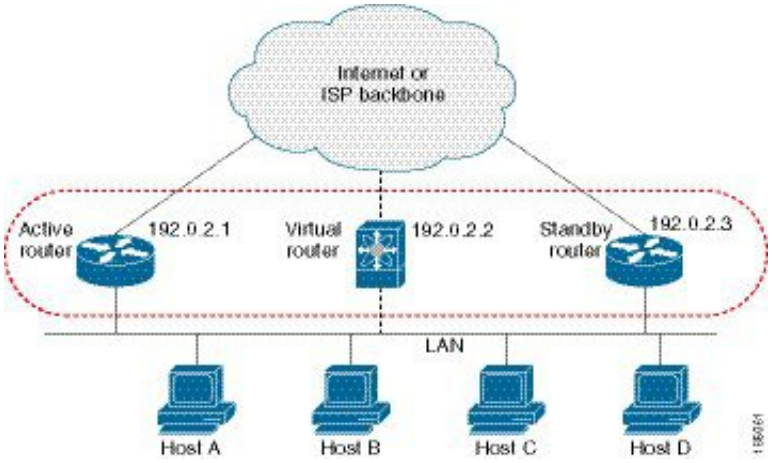
1. Configure the virtual IP and MAC address on all HSRP-enabled interfaces.
2. Assign a higher priority to the interface you want to be the default active router (default priority is 100).
3. HSRP selects the active and standby routers based on priority and hello message status.

- Active router: Forwards packets sent to the virtual IP/MAC address.
- Standby router: Monitors the active router and takes over if it fails.
- Virtual router: Represents the shared default gateway for hosts, even though it does not physically exist.



Note Packets destined for the HSRP virtual IP address on a routed port terminate on the local router, regardless of its HSRP role. On a Layer 2 (VLAN) interface, such packets terminate on the active router.

Figure 44: HSRP Topology with Two Enabled Routers



Example: HSRP in a Redundant Network

In a network with two routers configured for HSRP, both share a virtual IP and MAC address. Hosts are configured to use the virtual IP as their default gateway. If the active router fails to send hello messages within the configured time, the standby router automatically takes over, ensuring uninterrupted connectivity for hosts.

HSRP versions

- HSRP version 1 is supported by default; interfaces can be configured to use HSRP version 2.
- HSRP version 2 expands the group number range from 0–255 (version 1) to 0–4095.
- HSRP version 2 uses different multicast and MAC address ranges, supports MD5 authentication, and introduces a TLV packet format.

HSRP version 2 introduces several enhancements over version 1, including expanded group numbers, updated multicast and MAC addresses, MD5 authentication, and a new packet format.

- HSRP version 2 supports group numbers from 0 to 4095; version 1 supports 0 to 255.
- For IPv4, HSRP version 2 uses multicast address 224.0.0.102; for IPv6, FF02::66. HSRP version 1 uses 224.0.0.2.
- HSRP version 2 uses MAC address range 0000.0C9F.F000–0000.0C9F.FFFF for IPv4 and 0005.73A0.0000–0005.73A0.0FFF for IPv6. Version 1 uses 0000.0C07.AC00–0000.0C07.ACFF.
- HSRP version 2 adds support for MD5 authentication.
- HSRP version 2 uses a type-length-value (TLV) packet format; version 1 routers ignore version 2 packets.
- Changing the HSRP version reinitializes the group due to a new virtual MAC address.

HSRP for IPv4

HSRP for IPv4 is a protocol that enables routers to provide a virtual default gateway for hosts, ensuring high availability and redundancy on a network segment.

- Routers exchange HSRP hello packets using multicast addresses and UDP port 1985.
- Hosts use the HSRP virtual IP and MAC addresses as their default gateway.
- HSRP version 2 supports an expanded group number range and new multicast and MAC addresses.

How HSRP for IPv4 Works

HSRP routers communicate by exchanging hello packets to maintain the status of the active and standby routers. These packets are sent to specific multicast addresses and use UDP port 1985. The active router uses the HSRP virtual MAC address, while the standby router uses its interface MAC address. Hosts are configured to use the HSRP virtual IP address as their default gateway and resolve the associated virtual MAC address using ARP.

- HSRP hello packets are sent to 224.0.0.2 (version 1) or 224.0.0.102 (version 2).
- The active router sources hello packets from its configured IP and the HSRP virtual MAC address.
- The standby router sources hello packets from its configured IP and the interface MAC address (BIA).
- Hosts use ARP to resolve the HSRP virtual IP to the virtual MAC address.
- HSRP group number determines the virtual MAC address (e.g., 0000.0C07.ACxy for version 1, 0000.0C9F.F000–0000.0C9F.FFFF for version 2).

HSRP for IPv6

HSRP for IPv6 is a protocol that provides a virtual first hop for IPv6 hosts, enabling rapid failover to an alternate default router in the event of a failure.

- Offers faster switchover than standard IPv6 neighbor discovery, with failover in less than a second when using millisecond timers.
- Provides a virtual IPv6 link-local address for hosts as the default gateway.
- Uses a virtual MAC address and specific protocol parameters for group messaging and redundancy.

HSRP for IPv6 enhances the default router redundancy for IPv6 hosts by providing a virtual first hop and rapid failover capabilities.

- HSRP for IPv6 provides a much faster switchover to an alternate default router than the IPv6 ND protocol provides.
- When HSRP is configured on an IPv6 interface, periodic router advertisements (RAs) for the interface link-local address stop after a final RA with a router lifetime of zero is sent.
- IPv6 ND sends periodic RAs for the HSRP virtual IPv6 link-local address when the HSRP group is active, and these RAs stop after a final RA is sent with a router lifetime of 0 when the group leaves the active state.
- HSRP uses the virtual MAC address for active group messages only (hello, coup, and resign).

HSRP for IPv6 uses the following parameters:

- HSRP version 2
- UDP port 2029
- Virtual MAC address range from 0005.73A0.0000 through 0005.73A0.0FFF
- Multicast link-local IP destination address of FF02::66
- Hop limit set to 255

HSRP for IPv6 addresses

An HSRP IPv6 group is defined by a virtual MAC address derived from the HSRP group number and a virtual IPv6 link-local address, which is by default derived from the HSRP virtual MAC address. The default virtual MAC address is always used to form the virtual IPv6 link-local address, regardless of the actual virtual MAC address used by the group.

- Each HSRP IPv6 group has a virtual MAC address based on the group number.
- The virtual IPv6 link-local address is derived from the default virtual MAC address.
- The default virtual MAC address is always used to form the link-local address, even if the group uses a different MAC address.

HSRP for IPv6 uses specific MAC and IPv6 addresses for neighbor discovery and HSRP packets. The following table summarizes the MAC and IP addresses used for different packet types.

Table 42: HSRP and IPv6 ND Addresses

Packet	MAC Source Address	IPv6 Source Address	IPv6 Destination Address	Link-Layer Address Option
Neighbor solicitation (NS)	Interface MAC address	Interface IPv6 address	—	Interface MAC address
Router solicitation (RS)	Interface MAC address	Interface IPv6 address	—	Interface MAC address
Neighbor advertisement (NA)	Interface MAC address	Interface IPv6 address	Virtual IPv6 address	HSRP virtual MAC address
Route advertisement (RA)	Interface MAC address	Virtual IPv6 address	—	HSRP virtual MAC address
HSRP (inactive)	Interface MAC address	Interface IPv6 address	—	—
HSRP (active)	Virtual MAC address	Interface IPv6 address	—	—

HSRP does not add IPv6 link-local addresses to the Unicast Routing Information Base (URIB). Link-local addresses do not have secondary virtual IP addresses. For global unicast addresses, HSRP adds the virtual IPv6 address to the URIB and IPv6.

Multiple group optimization for HSRP

Beginning with Cisco NX-OS Release 7.0(3)I7(1), HSRP supports multiple group optimization (MGO). MGO optimizes performance and bandwidth when multiple HSRP groups are configured on many subinterfaces. MGO requires only one HSRP group, known as the master group, on the physical interface for the purpose of electing active and standby routers.

Administrators can create other HSRP groups on subinterfaces of the physical interface or a different interface, such as an SVI interface, and link these to the master HSRP group. These groups are known as slave groups. Slave groups follow their master group state so that they do not participate in any HSRP election mechanisms. Master groups send hello messages at their configured rates. Slave groups send hello messages at a reduced rate, which is called the mac-refresh interval rate. This process is required so that the slave groups can send out periodic messages in order to refresh MAC addresses in switches and learning bridges.

HSRP subnet VIP

An HSRP subnet virtual IP (VIP) is a virtual IP address that can be configured in a different subnet than the interface IP address.

- Enables conservation of public IPv4 addresses by using a VIP as a public IP and an interface IP as a private IP.
- Allows periodic ARP synchronization to vPC peers and ARP sourcing with the VIP when configured for hosts in the VIP subnet.
- Not required for IPv6 addresses due to the larger address pool and routable IPv6 addresses on an SVI with regular HSRP.

You can configure HSRP subnet VIPs for Cisco Nexus 9508 platform switches with the 9636C-R, 9636C-RX, and 9636Q-R line cards.

- For more information, see Guidelines and Limitations for HSRP and Configuration Examples for HSRP .

HSRP authentication

HSRP authentication is a security mechanism that uses the MD5 algorithm to protect HSRP messages from spoofing and ensures message integrity by including the IPv4 or IPv6 address in the authentication TLVs.

- Protects against HSRP-spoofing software.
- Uses the industry-standard MD5 algorithm for authentication.
- Includes IPv4 or IPv6 address in authentication TLVs for improved reliability and security.

HSRP messages

- HSRP messages are multicast messages exchanged between routers configured with HSRP.

- They convey HSRP priority and state information, and manage the transition between active and standby routers.
- Key message types include Hello, Coup, and Resign.

Routers configured with HSRP exchange the following 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 load sharing is a method that allows multiple HSRP groups to be configured on an interface, enabling traffic from connected hosts to be distributed across multiple routers while maintaining default router redundancy.

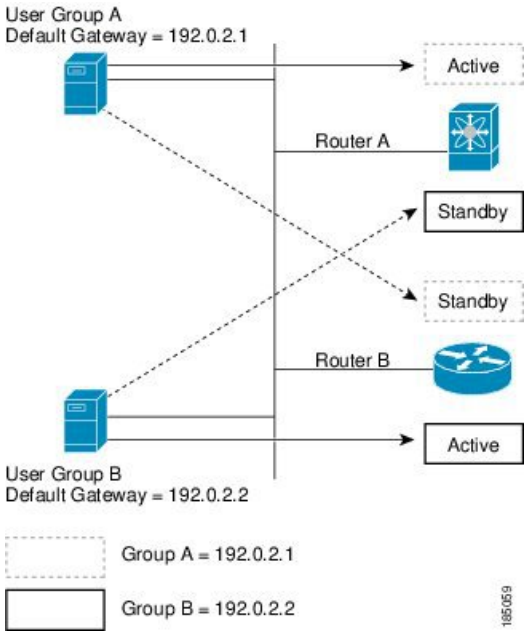
- Supports configuration of multiple, overlapping HSRP groups on a single interface.
- Enables load balancing of host traffic across two or more routers.
- Maintains default router redundancy in case of router failure.

HSRP load sharing is achieved by configuring two overlapping IPv4 HSRP groups on an interface. Each router is active for one group and standby for the other, allowing traffic to be balanced between them. If one router fails, the other takes over both groups, ensuring continued connectivity.



Note HSRP for IPv6 load balances by default. If two HSRP IPv6 groups are on the subnet, hosts learn of both groups from their router advertisements and choose to use one so that the load is shared between the advertised routers.

Figure 45: HSRP Load Sharing



Example of HSRP load sharing configuration

For example, in a network with two routers (A and B) and two HSRP groups, Router A is the active router for group A and standby for group B, while Router B is active for group B and standby for group A. This setup allows host traffic to be balanced across both routers. If either router fails, the remaining router processes traffic for both hosts.

Object tracking and HSRP

Object tracking is a mechanism that enables the modification of HSRP interface priority based on the operational state of another interface.

- Allows dynamic adjustment of HSRP priority.
- Supports tracking of the line protocol state of an interface or the reachability of an IP route.
- Facilitates routing to a standby router if the main network interface fails.

You can use object tracking to modify the priority of an HSRP interface based on the operational state of another interface. If the specified object goes down, Cisco NX-OS reduces the HSRP priority by the configured amount.

Two objects that you can track are the line protocol state of an interface or the reachability of an IP route.

- Line protocol state of an interface
- Reachability of an IP route

For more information, see the Configuring HSRP Object Tracking section.

vPCs and HSRP

vPCs and HSRP integration enables redundancy and load balancing by allowing both the active and standby HSRP routers to forward traffic through physically separate switches that appear as a single logical port channel.

- vPCs aggregate links from two Cisco Nexus 9000 Series switches to present as a single port channel.
- HSRP provides active and standby router roles for high availability.
- Both active and standby HSRP routers can forward traffic in a vPC topology.

How vPCs and HSRP Interoperate

HSRP interoperates with vPCs by allowing both the active and standby HSRP routers to forward traffic, providing redundancy and efficient traffic distribution.

- Links from two Nexus 9000 Series switches appear as a single port channel to a third device.
- HSRP active role can be distributed across primary and secondary vPC peers for different SVIs.

1. Configure vPC between two Nexus 9000 Series switches.
2. Enable HSRP on the relevant SVIs.
3. Verify that both active and standby HSRP routers can forward traffic.

- vPC: Aggregates links from two switches.
- HSRP: Provides redundancy and failover.

For more information, see the Cisco Nexus 9000 Series NX-OS Layer 2 Switching Configuration Guide , Configuring the HSRP Priority section, and Configuration Examples for HSRP section.



Note HSRP active can be distributed on both the primary and secondary vPC peers for different SVIs.

vPC peer gateway and HSRP

The vPC peer gateway is a feature that enables HSRP routers in a vPC environment to process packets addressed to the local vPC peer MAC address, the remote vPC peer MAC address, and the HSRP virtual MAC address directly.

- Prevents third-party devices from causing packets to be sent across the vPC peer link using the source MAC address of an HSRP router.
- Ensures packets addressed to the HSRP virtual MAC or vPC peer MAC addresses are handled locally by the correct router.
- Reduces the risk of potential packet drops in vPC environments with HSRP.

Some third-party devices can ignore the HSRP virtual MAC address and instead use the source MAC address of an HSRP router. In a vPC environment, this behavior can cause packets to be sent across the vPC peer link, potentially resulting in dropped packets. Configuring the vPC peer gateway allows HSRP routers to directly handle these packets.

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

BFD

BFD is a detection protocol that provides subsecond failure detection between two adjacent devices and can be less CPU-intensive than protocol hello messages.

- Provides fast-forwarding and path-failure detection times.
- Enables subsecond failure detection between adjacent devices.
- Can offload some processing to the data plane on supported modules, reducing CPU usage.

BFD is supported for HSRP 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.

For more information, see the Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide .

High availability and extended nonstop forwarding

- HSRP supports stateful restarts, which occur when the HSRP process fails and is restarted.
- HSRP supports stateful switchovers, which occur when the active supervisor switches to the standby supervisor.
- Extended nonstop forwarding (NSF) allows HSRP to temporarily extend hold timers during a controlled switchover, preventing unnecessary state changes.

HSRP uses extended NSF to ensure continuous operation during supervisor switchovers by managing hold timers and hello messages.

- HSRP applies the run-time configuration after a switchover.
- When HSRP hold timers are configured for short periods, they might expire during a controlled switchover.
- HSRP sends hello messages with extended timers when extended NSF is configured.
- HSRP peers update their hold timers with the new extended values.
- Extended timers prevent unnecessary HSRP state changes during the switchover.
- After the switchover, HSRP restores the hold timers to their original configured values.
- If the switchover fails, HSRP restores the hold timers after the extended hold timer values expire.

For more information, see the Configuring Extended Hold Timers for HSRP section.

Virtualization support

Virtualization support refers to the capability of HSRP to operate within virtual routing and forwarding (VRF) instances. HSRP can be configured to support multiple VRF instances.

Prerequisites for HSRP

You must enable the HSRP feature in a device before you can configure and enable any HSRP groups.

Guidelines and limitations for HSRP

Configure an IP address for the interface that you configure HSRP on and enable that interface before HSRP becomes active.

Cisco Nexus 9500 platform switches running in max-host routing mode do not support four-way HSRP.

Configure HSRP version 2 when you configure an IPv6 interface for HSRP.

For IPv4, the virtual IP address must be in the same subnet as the interface IP address.

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 allowed group number range for version 1 (0-255).

HSRP for IPv4 is supported with BFD. HSRP for IPv6 is not supported with BFD.

If HSRP IPv4 and IPv6 use the same virtual MAC address on an SVI, the HSRP state must be the same for both HSRP IPv4 and IPv6. The priority and preemption should be configured to result in the same state after failovers.

Cisco NX-OS removes all Layer 3 configurations on an interface when you change the interface VRF membership, port channel membership, or the port mode to Layer 2.

If you configure virtual MAC addresses with vPC, you must configure the same virtual MAC address on both vPC peers.

You cannot use the HSRP MAC address burned-in option on a VLAN interface that is a vPC member.

Cisco NX-OS supports having the same HSRP groups on all nodes in a double-sided vPC.

If you have not configured authentication, the **show hsrp** command displays the following string:

```
Authentication text "cisco"
```

The default behavior of HSRP is as defined in RFC 2281:

```
If no authentication data is configured, the RECOMMENDED default
     value is 0x63 0x69 0x73 0x63 0x6F 0x00 0x00 0x00.
```

When configuring 4-way HSRP using 2 pairs of vPC switches (new deployment or migration scenarios), the HSRP priorities should be configured such that the vPC pairs of Nexus 9000 switches are in Active/Standby

state and Listen/Listen state. There is no support for Cisco Nexus 9000 vPC peers to be in HSRP Active/Listen state, or Standby/Listen state.

HSRP subnet VIP feature guidelines and limitations

The HSRP subnet VIP feature has the following guidelines and limitations:

- This feature is supported for Cisco Nexus 9000 Series switches and for Cisco Nexus 9508 switches with the 9636C-R, 9636C-RX, and 9636Q-R line cards.
- 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.
- Unicast Reverse Path Forwarding (URPF) and DHCP sourcing with VIPs are not supported with this feature.
- 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 continues to source with SVI IP addresses and not with the VIP.
- If the subnet VIP is configured with /32 as the length, you must use the **no** command with /32 to remove the IP address (for example, **no ip ip-address/32**).

To remove an SVI configuration with its sub-configurations, that are configured using a configuration profile, you must first remove the profile or clear the manual configuration settings under the VLAN before executing **no interface vlan** command.

HSRP for MGO has the following limitations:

- Master groups and slave groups are not restricted to the same interface.
- HSRP for MGO supports only HSRP version 2.
- Master and slave groups must have the same address types.
- Configuring an HSRP group as a slave group clears the group's other configurations, such as its virtual IP address, without notification, so you must enter the follow command before you enter the ipip-address command.
- Bidirectional forwarding (BFD) is not applicable to slave groups.
- HSRP for MGO supports both IPv4 and IPv6 interfaces and works for all Layer 3 interfaces on which a regular HSRP group works.
- An HSRP group cannot be configured as both a master and slave group at the same time.
- Group cannot be configured as master on unnumbered interface.
- Few group commands such as priority, preempt, timers, authentication and IP learn mode are not applicable to slave groups.

Configuration guidelines to enforce the pre-empt reload timer

The following are configuration guidelines to enforce the pre-empt reload timer. The guidelines are listed in order of decreasing preference.

1. In triangle topologies, we recommend that the HSRP peers are configured within a single VPC domain. This configuration prevents the Spanning-Tree root bridge from changing on the HSRP peer when the Cisco Nexus 9000 configuration is reloaded.
2. Make sure the Spanning Tree root bridge for all VLANs is not on the Cisco Nexus 9000 that is being reloaded.
3. If 1 and 2 are not possible, make sure that the switch has an enabled link for all the SVI VLANs that is connected to another switch that is not the HSRP peer.

Beginning with Cisco NX-OS Release 10.6(1)F, Unicast Reverse Path Forwarding (URPF), HSRP, vPC, and vPC peer gateway are supported on Cisco Nexus N9336C-SE1 switches.

Beginning with Cisco NX-OS Release 10.6(1)F, user-defined MAC address is supported on Cisco N9336C-SE1 switches.



Note For Nexus N9336C-SE1 switches, of the total Router MAC scale, the number of unique 32 bit MSBs allowed for user-defined MAC addresses is 11.

Beginning with Cisco NX-OS Release 10.6(2)F, HSRP and uRPF are supported on Cisco Nexus N9324C-SE1U and N9348Y2C6D-SE1U platform switches.

Default settings for HSRP parameters

This topic provides the default values for HSRP parameters.

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

Configue HSRP

Enable HSRP

You must globally enable HSRP before you can configure and enable any HSRP groups.

Procedure

[**no**] **feature hsrp**

Example:

```
switch(config)# feature hsrp
```

Enables the HSRP feature. Use the **no** form of this command to disable HSRP for all groups.

Configure 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.

Note IPv6 HSRP groups must be configured as HSRP version 2.

Procedure

hsrp version {1 | 2}

Example:

```
switch(config-if)# hsrp version 2
```

Confirms the HSRP version. Version 1 is the default.

Configure an HSRP group for IPv4

Use this task to configure an HSRP group for IPv4 on an interface.

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 HSRP section).

Cisco NX-OS enables an HSRP group once you configure the virtual IP address. You must configure HSRP attributes such as authentication, timers, and priority before you enable the HSRP group.

Procedure

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 **ip** *ip-address/length*

Example:

```
switch(config-if)# ip 192.0.2.2/8
```

Configures the IPv4 address of the 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. The range for HSRP version 1 is from 0 to 255. The range for HSRP version 2 is from 0 to 4095. The default value is 0.

Step 5 **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 6 **exit**

Example:

```
switch(config-if-hsrp)# exit
```

Exits HSRP configuration mode.

Step 7 **no shutdown**

Example:

```
switch(config-if-hsrp)# no shutdown
```

Enables the interface.

Step 8 (Optional) **show hsrp** [**group** *group-number*] [**ipv4**]

Example:

```
switch(config-if-hsrp)# show hsrp group 2
```

Displays HSRP information.

Step 9 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if-hsrp)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

After completing these steps, the HSRP group is configured and enabled on the interface.



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)# 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
```

Configure an HSRP group for IPv6

Use this task to configure an HSRP group for IPv6 on an interface.

You can configure an HSRP group on an IPv6 interface and configure the virtual MAC address for the HSRP group.

When you configure an HSRP group for IPv6, HSRP generates a link-local address from the link-local prefix. HSRP also generates a modified EUI-64 format interface identifier in which the EUI-64 interface identifier is created from the relevant HSRP virtual MAC address.

Before you begin

You must enable HSRP (see the Enabling HSRP section).

Ensure that you have enabled HSRP version 2 on the interface on which you want to configure an IPv6 HSRP group.

Ensure that you have configured HSRP attributes such as authentication, timers, and priority before you enable the HSRP group.

Procedure

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 3/2
switch(config-if)#
```

Enters interface configuration mode.

Step 3 **ipv6 address** *ipv6-address/length*

Example:

```
switch(config-if)# ipv6 address 2001:0DB8::0001:0001/64
```

Configures the IPv6 address of the interface.

Step 4 **hsrp version 2**

Example:

```
switch(config-if-hsrp)# hsrp version 2
```

Configures the group for HSRP version 2.

Step 5 **hsrp** *group-number* **ipv6**

Example:

```
switch(config-if)# hsrp 10 ipv6
switch(config-if-hsrp)#
```

Creates an IPv6 HSRP group and enters HSRP configuration mode. The range for HSRP version 2 is from 0 to 4095. The default value is 0.

Step 6 **ip** *ipv6-address*

Example:

```
switch(config-if-hsrp)# ip 2001:DB8::1
```

Configures the virtual IPv6 address for the HSRP group and enables the group.

Step 7 **ip autoconfig**

Example:

```
switch(config-if-hsrp)# ip autoconfig
```

Autoconfigures the virtual IPv6 address for the HSRP group from the calculated link-local virtual IPv6 address and enables the group.

Step 8 **exit**

Example:

```
switch(config-if-hsrp)# exit
switch(config-if)#
```

Exits HSRP configuration mode.

Step 9 **no shutdown**

Example:

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

Enables the interface.

Step 10 (Optional) **show hsrp** [**group** *group-number*] [**ipv6**]

Example:

```
switch(config-if)# show hsrp group 10
```

Displays HSRP information.

Step 11 **copy running-config startup-config**

Example:

```
switch(config-if)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.



Note You should use the **no shutdown** command to enable the interface after you finish the configuration.

This example shows how to configure an IPv6 HSRP group on Ethernet 3/2:

```
switch# configure terminal
switch(config)# interface ethernet 3/2
switch(config-if)# ipv6 address 2001:0DB8::0001:0001/64
switch(config-if-hsrp)# hsrp version 2
switch(config-if)# hsrp 2 ipv6
switch(config-if-hsrp)# ip 2001:DB8::1
switch(config-if-hsrp)# exit
switch(config-if)# no shutdown
switch(config-if)# copy running-config startup-config
```

Configure an HSRP master group task

Set up an hsrp master group task for the selected deployment workflow.

Follow these steps to configure an hsrp master group task:

Before you begin

- Ensure that you have enabled the HSRP feature.

- Configure HSRP attributes such as authentication, timers, and priority before you enable an HSRP group as a master group.

Procedure

Step 1 Configure the interface and IP address.

a) switch# **configure terminal**

Enters global configuration mode.

b) switch(config)# **interface** *type/number*

Enters interface configuration mode and configures an interface type.

c) switch(config-if)# **ip address***ip-address/length*

Configures the IP address of the interface.

Step 2 Configure the HSRP version and master group.

a) switch(config-if)# **hsrp version 2**

Configures the HSRP version. Because MGO supports only HSRP version 2, you must set the HSRP version to version 2. Version 1 is the default.

b) switch(config-if)# **hsrp** *group-number* [**ipv6**]

Creates an HSRP group and enters HSRP configuration mode. The range for the HSRP group number is from 0 to 4095. The no form of this command removes the group.

c) switch(config-if-hsrp)# **name** [*master-group-name*]

Specifies a master group name. The name command changes a regular HSRP group into a master group. If you do not specify a name, a unique name is automatically generated. The no form of this command returns the master group to a regular HSRP group.

Step 3 Configure the virtual IP address and enable the interface.

a) switch(config-if-hsrp)# **ip** [*ip-address* [**secondary**]]

Configures the virtual IP address for the HSRP group and enables the master group.

b) switch(config-if-hsrp)# **exit**

Exits the HSRP configuration mode.

c) switch(config-if)# **no shutdown**

Enables the interface.

Step 4 Verify and save the configuration.

a) switch(config-if)# **show hsrp** [**brief**] [**group** *group-number*] [**ipv4**] [**ipv6**]

(Optional)

Displays HSRP information.

b) switch(config-if)# **show hsrp mgo** [**name** *name*] [**brief**]

(Optional)

Displays the relationships between HSRP groups that are in use for MGO and their slave sessions. The name keyword restricts the output to the session with a matching configured name. The brief keyword provides a summary of each MGO session with the associated slave sessions.

The an hsrp master group task task is complete.

This example shows how to configure an HSRP master group on Ethernet interface 1/1:

```
switch# configure terminal
switch(config)# interface ethernet 1/1
switch(config-if)# ip address 192.0.2.0/24
switch(config-if)# hsrp version 2
switch(config-if)# hsrp 11
switch(config-if-hsrp)# name master1
switch(config-if-hsrp)# ip 192.0.2.1
switch(config-if-hsrp)# exit
switch(config-if)# no shutdown
switch# show hsrp group 11
switch# show hsrp mgo name master1
```

Configure an HSRP slave group

Set up an hsrp slave group for the selected deployment workflow.

Follow these steps to configure an hsrp slave group:

Before you begin

- Ensure that you have enabled the HSRP feature.
- Configure HSRP attributes such as authentication, timers, and priority before you enable an HSRP group as a master group.

Procedure

Step 1 Configure the interface and IP address.

a) switch# **configure terminal**

Enters global configuration mode.

b) switch(config)# **interface** *type/number*

Enters interface configuration mode and configures an interface type.

c) switch(config-if)# **ip address***ip-address/length*

Configures the IP address of the interface.

Step 2 Configure the HSRP version and group.

a) switch(config-if)# **hsrp version 2**

Configures the HSRP version. Because MGO supports only HSRP version 2, you must set the HSRP version to version 2. Version 1 is the default.

b) switch(config-if)# **hsrp mac-refresh** *seconds*

(Optional)

Configures the MAC refresh interval for the HSRP slave group. Administrators can use this command to minimize the number of hello messages that are sent out and reduce HSRP protocol overheads and CPU utilization when multiple subinterfaces are configured.

This command is not available for individual subinterfaces. It applies to all groups on all subinterfaces. The default is 60 seconds. The range is from 0 to 10000.

c) switch(config-if)# **hsrp** *group-number* [**ipv6**]

Creates an HSRP group and enters HSRP configuration mode. The range for the HSRP group number is from 0 to 4095. The no form of this command removes the group.

Step 3 Configure the slave group and virtual IP address.

a) switch(config-if-hsrp)# **follow** [*master-group-name*]

Configures a regular HSRP group as a slave group.

Configuring an HSRP group as a slave group clears the group's other configurations, such as its virtual IP address without notification, so you must enter the follow command before you enter the ip ip-address command.

Slave groups may forward reference master group names that are undefined.

The no form of this command returns the slave group to a regular HSRP group.

b) switch(config-if-hsrp)# **ip** [*ip-address*]

Configures the virtual IP address for the HSRP group and enables the slave group.

c) switch(config-if-hsrp)# **exit**

Exits the HSRP configuration mode.

Step 4 Enable and verify the slave group.

a) switch(config-if)# **no shutdown**

Enables the interface.

b) switch(config-if)# **show hsrp** [**brief**] [**group** *group-number*] [**ipv4**] [**ipv6**]

(Optional)

Displays HSRP information.

c) switch(config-if)# **show hsrp mgo** [**name** *name*] [**brief**]

(Optional)

Displays the relationships between HSRP groups that are in use for MGO and their slave sessions. The name keyword restricts the output to the session with a matching configured name. The brief keyword provides a summary of each MGO session with the associated slave sessions.

The an hsrp slave group task is complete.

This example shows how to configure an HSRP slave group on Ethernet interface 1/2:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ip address 198.51.100.0/24
switch(config-if)# hsrp version 2
switch(config-if)# hsrp 12
switch(config-if-hsrp)# follow master1
switch(config-if-hsrp)# ip 192.0.2.2
switch(config-if-hsrp)# exit
switch(config-if)# no shutdown
switch# show hsrp group 11
switch# show hsrp mgo name master1
```

Configure the HSRP virtual MAC address

You can override the default virtual MAC address that HSRP derives from the configured group number.

Note You must configure the same virtual MAC address on both vPC peers of a vPC link.

Procedure

Step 1 **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).

Step 2 (Optional) **hsrp use-bia [scope interface]**

Example:

```
switch(config-if)# hsrp use-bia
```

Note 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:

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** keyword.

Authenticate HSRP

Use this task to configure authentication for HSRP, ensuring secure protocol operation on your network devices.

You can configure HSRP to authenticate the protocol using cleartext or MD5 digest authentication. MD5 authentication uses a keychain. For more details, see the Cisco Nexus 9000 Series NX-OS Security Configuration Guide .

Before you begin

You must enable HSRP (see the Enabling HSRP section).

Ensure that you have configured the same authentication and keys on all members of the HSRP group.

Ensure that you have created the keychain if you are using MD5 authentication.

Procedure

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 **hsrp** *group-number* [**ipv4** | **ipv6**]

Example:

```
switch(config-if)# hsrp 2
switch(config-if-hsrp)#
```

Creates an HSRP group and enters HSRP configuration mode.

Step 4 **authentication** { **text** *string* | **md5** { **key-chain** *key-chain* | **key-string** { **0** | **7** } *text* [**compatibility**] [**timeout** *seconds*]}}

Example:

```
switch(config-if-hsrp)# authentication text mypassword
```

Example:

```
switch(config-if-hsrp)# authentication md5 key-chain hsrp-keys
```

Configures cleartext authentication for HSRP on this interface using the **authentication text** command or configures MD5 authentication for HSRP on this interface using the **authentication md5** command.

If you configure MD5 authentication, you can use a keychain or key string. If you use a key string, you can optionally set the timeout for when HSRP only accepts a new key. The range is from 0–32,767 seconds.

Compatibility: Designed for authentication compatibility between Cisco IOS and Cisco NX-OS. Compatibility mode is for MD5 key-string authentication. When a hidden authentication type is configured on both Cisco IOS and Cisco NX-OS, the compatibility flag has to be enabled in NX-OS to bring up the HSRP session.

Step 5 (Optional) **show hsrp** [**group** *group-number*]

Example:

```
switch(config-if-hsrp)# show hsrp group 2
```

Displays HSRP information.

Step 6 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if-hsrp)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

HSRP authentication is configured. The device will use the specified authentication method for HSRP group communication.

This example shows how to configure MD5 authentication for HSRP on Ethernet 1/2 after creating the keychain:

```
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 2013 23:59:59 Sep 12 2013
switch(config-keychain-key) send-lifetime 00:00:00 Jun 01 2013 23:59:59 Aug 12 2013
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 2013 23:59:59 Dec 12 2013
switch(config-keychain-key) send-lifetime 00:00:00 Sep 12 2013 23:59:59 Nov 12 2013
switch(config-keychain-key)# interface ethernet 1/2
switch(config-if)# hsrp 2
switch(config-if-hsrp)# authentication md5 key-chain hsrp-keys
switch(config-if-hsrp)# copy running-config startup-config
```

Configure 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 an HSRP group 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.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **track** *object-id* **interface** *interface-type slot/port* { **line-protocol** | **ip routing** | **ipv6 routing** }

Example:

```
switch(config)# track 1 interface ethernet 2/2 line-protocol
switch(config-track)#
```

Configures the interface that the track object tracks. Changes in the state of the interface affect the track object status as follows:

- You configure the interface and corresponding object number that you use with the **track** command in global configuration mode.
- The **line-protocol** keyword tracks whether the interface is up. The **ip routing** or **ipv6 routing** keyword also checks that IP routing is enabled on the interface and an IP address is configured.

Step 3 **track** *object-id* { **ip** | **ipv6** } **route** *ip-prefix/length* **reachability**

Example:

```
switch(config-track)# track 2 ip route 192.0.2.0/8 reachability
```

Creates a tracked object for a route and enters tracking configuration mode. The *object-id* range is from 1 to 500.

Step 4 **exit**

Example:

```
switch(config-track)# exit
switch(config)#
```

Exits track configuration mode.

Step 5 **interface** *interface-type slot/port*

Example:

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

Enters interface configuration mode.

Step 6 **hsrp** *group-number* [**ipv4** | **ipv6**]

Example:

```
switch(config-if)# hsrp 2
switch(config-if-hsrp)#
```

Creates an HSRP group and enters HSRP configuration mode.

Step 7 **priority** [*value*]

Example:

```
switch(config-if-hsrp)# priority 254
```

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.

Step 8 **track** *object-id* [**decrement** *value*]

Example:

```
switch(config-if-hsrp)# track 1 decrement 20
```

Specifies an object to be tracked that affects the weighting of an HSRP interface.

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 9 **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. Optionally, a delay can be configured that delays the HSRP group preemption by the configured time. The range is from 0 to 3600 seconds.

Step 10 (Optional) **show hsrp interface** *interface-type slot/port*

Example:

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

Displays HSRP information for an interface.

Step 11 **copy running-config startup-config**

Example:

```
switch(config-if-hsrp)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

This example shows how to configure HSRP object tracking on Ethernet interface 1/2:

```
switch# configure terminal
switch(config)# track 1 interface ethernet 2/2 line-protocol
switch(config-track)# track 2 ip route 192.0.2.0/8 reachability
switch(config-track)# exit
switch(config)# interface ethernet 1/2
switch(config-if)# hsrp 2
switch(config-if-hsrp)# priority 254
switch(config-if-hsrp)# track 1 decrement 20
switch(config-if-hsrp)# preempt delay minimum 60
switch(config-if-hsrp)# copy running-config startup-config
```

Configure the HSRP priority

You can configure the priority of an HSRP group. HSRP uses the priority to determine which HSRP group member acts as the active router. If you configure HSRP on a vPC-enabled interface, you can optionally configure the upper and lower threshold values to control when to fail over to the vPC trunk. If the standby router priority falls below the lower threshold, HSRP sends all standby router traffic across the vPC trunk to forward through the active HSRP router. HSRP maintains this scenario until the standby HSRP router priority increases above the upper threshold.

For IPv6 HSRP groups, if all group members have the same priority, HSRP selects the active router based on the IPv6 link-local address.

To configure the HSRP priority, use the following command in the HSRP group configuration mode:

Procedure

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. Optionally, this command sets the upper and lower threshold values used by vPC to determine when to fail over to the vPC trunk. The *lower-value* range is from 1 to 255. The default is 1. The *upper-value* range is from 1 to 255. The default is 255.

Customize HSRP in HSRP configuration mode

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 becomes operational. If you 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.

Procedure

Step 1 (Optional) **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.

Step 2 (Optional) **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. Optionally, a delay can be configured that delays the HSRP group preemption by the configured time. The range is from 0 to 3600 seconds.

Step 3 (Optional) **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:

- *hellotime* —The interval between successive hello packets sent. The range is from 1 to 254 seconds.
- *holdtime* —The interval before the information in the hello packet is considered invalid. The range is from 3 to 255.

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 250 to 999 milliseconds.
- *holdtime* —The interval before the information in the hello packet is considered invalid. The range is from 750 to 3000 milliseconds.

Step 4 (Optional) **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.

Step 5 (Optional) **hsrp delay reload** *seconds*

Example:

```
switch(config-if)# hsrp delay reload 30
```

Specifies the minimum amount of time that HSRP waits after a reload and before participating in the group. The range is from 0 to 10000 seconds. The default is 0.

Note When using preempt delay with 'reload' option, the recommendation is to use it along with **hsrp delay reload** (interface-level command). This is to avoid the scenario where after reload, higher priority HSRP Standby becomes Active on hold timer expiry (10 seconds) because the preempt delay reload timer didn't start as SVI is UP but the physical link/port-channel is not yet UP after reload. Timers can be tuned according to scale.

Example - Instead of configuring **preempt delay reload 200** , configure **preempt delay reload 140** and **hsrp delay reload 60** . This is to ensure that the SVI and physical link/port-channel are both UP, when HSRP starts the start machine from INIT state after reload delay expiry (60 sec).

Customize HSRP in interface configuration mode

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 becomes operational. If you 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.

Procedure

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 **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.

Step 4 **hsrp delay reload** *seconds*

Example:

```
switch(config-if)# hsrp delay reload 30
```

Specifies the minimum amount of time that HSRP waits after a reload and before participating in the group. The range is from 0 to 10000 seconds. The default is 0.

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

Example:

```
switch(config-if)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

Configure extended hold timers for HSRP

You can configure HSRP to use extended hold timers to support extended NSF during a controlled (graceful) switchover. You should configure extended hold timers on all HSRP routers.

Note You must configure extended hold timers on all HSRP routers if you configure extended hold timers. If you configure a nondefault hold timer, you should configure the same value on all HSRP routers when you configure HSRP extended hold timers.

Note HSRP extended hold timers are not applied if you configure millisecond hello and hold timers for HSRPv1. This statement does not apply to HSRPv2.

Procedure

Step 1 (Optional) **hsrp timers extended-hold** [*timer*]

Example:

```
switch(config)# hsrp timers extended-hold
```

Sets the HSRP extended hold timer in seconds for both IPv4 and IPv6 groups. The *timer* range is from 10 to 255. The default is 10.

Note Use the **show hsrp** command or the **show running-config hsrp** command to display the extended hold time.

Step 2 (Optional) **show hsrp**

Example:

```
switch(config)# show hsrp
```

Displays the HSRP extended hold time.

Use the **show hsrp** command or the **show running-config hsrp** command to display the extended hold time.

Verify the HSRP Configuration

To display 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.

Command	Purpose
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.
show ip local-pt	Displays whether the netstack has programmed a subnet route for the VIP subnet.

Configuration examples for HSRP

- HSRP can be enabled on interfaces with MD5 authentication and interface tracking.
- HSRP priority can be configured and adjusted using the **priority** and **track** commands.
- HSRP subnet VIP addresses must not be in the same subnet as the interface IP address to avoid configuration errors.

HSRP Configuration Reference

These examples demonstrate how to configure HSRP with authentication, interface tracking, priority, and VIP address settings.

Examples

Enable HSRP with MD5 authentication and interface tracking:

```
key chain hsrp-keys
key 0
key-string 7 zqdest
accept-lifetime 00:00:00 Jun 01 2013 23:59:59 Sep 12 2013
send-lifetime 00:00:00 Jun 01 2013 23:59:59 Aug 12 2013
key 1
key-string 7 uaeqdyito
accept-lifetime 00:00:00 Aug 12 2013 23:59:59 Nov 12 2013
send-lifetime 00:00:00 Sep 12 2013 23:59:59 Nov 12 2013
feature hsrp
track 2 interface ethernet 2/2 ip
interface ethernet 1/2
ip address 192.0.2.2/8
hsrp 1
authenticate md5 key-chain hsrp-keys
priority 90
track 2 decrement 20
ip 192.0.2.10
no shutdown
```

Configure HSRP priority on an interface:

```
interface vlan 1
hsrp 0
preempt
priority 100 forwarding-threshold lower 80 upper 90
ip 192.0.2.2
track 1 decrement 30
```

Configure an HSRP subnet VIP address in a different subnet than the interface IP address:

```
sswitch# 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
```

Example of VIP subnet mismatch error:

```
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!
```

Example of error when VIP is 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

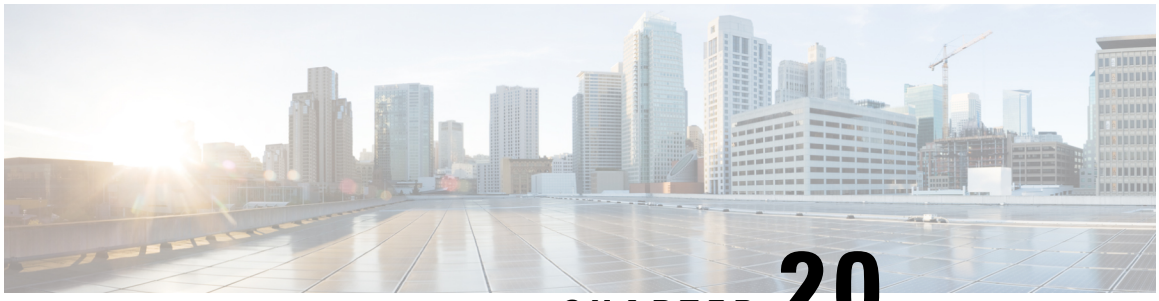
This section provides references to related documents for configuring VRRP and high availability.

Related Topic	Document Title
Configuring the Virtual Router Redundancy Protocol	Configuring VRRP
Configuring high availability	Cisco Nexus 9000 Series NX-OS High Availability and Redundancy Guide

MIBs

This section provides information about MIBs related to HSRP and where to locate and download them.

MIBs	MIBs Link
MIBs related to HSRP	To locate and download supported MIBs, go to the following URL: https://cisco.github.io/cisco-mibs/supportlists/nexus9000/Nexus9000MIBSupportList.html



CHAPTER 20

Configure VRRP

This chapter contains the following sections:

- About VRRP, on page 643
- Information about VRRPv3 and VRRS, on page 651
- High availability, on page 652
- Virtualization support, on page 652
- Guidelines and limitations for VRRP, on page 652
- Guidelines and limitations for VRRPv3, on page 653
- Default settings for VRRP parameters, on page 653
- Default settings for VRRPv3 parameters, on page 654
- Configure VRRP, on page 654
- Configure VRRPv3, on page 665
- Verify the VRRP configuration, on page 672
- Verify the VRRPv3 configuration, on page 672
- Monitor and clear VRRP statistics, on page 673
- Monitor and clear VRRPv3 statistics, on page 673
- Configuration examples for VRRP, on page 673
- Configuration examples for VRRPv3, on page 674
- Additional references, on page 676

About VRRP

VRRP (Virtual Router Redundancy Protocol) is a protocol that allows a group of routers to share a virtual IP address, providing transparent failover at the first-hop IP router.

- Configures a group of routers to share a virtual IP address.
- Elects one router in the group to handle all packets for the virtual IP address.
- Other routers remain in standby and take over if the active router fails.

VRRP operation

VRRP (Virtual Router Redundancy Protocol) is a protocol that allows multiple routers to form a group and share a single virtual IP address, which is used as the default gateway for LAN clients.

- Enables a group of routers (VRRP group) to share a virtual IP address.
- Provides redundancy for the default gateway, ensuring continuous network access if the primary router fails.
- Allows LAN clients to be configured with a single default gateway IP address, simplifying configuration and improving reliability.

How VRRP operation provides gateway redundancy for LAN clients

LAN clients can determine their first-hop router to a remote destination using either dynamic discovery protocols or static configuration. Dynamic methods include Proxy ARP, routing protocols, and ICMP Router Discovery Protocol (IRDP). However, these methods can introduce configuration and processing overhead, and may result in slow failover if a router becomes unavailable. Static configuration of a default router simplifies client setup but creates a single point of failure, potentially isolating the client from the network if the default gateway fails. VRRP addresses this by allowing multiple routers to share a virtual IP address, which clients use as their default gateway, thus providing redundancy and seamless failover.

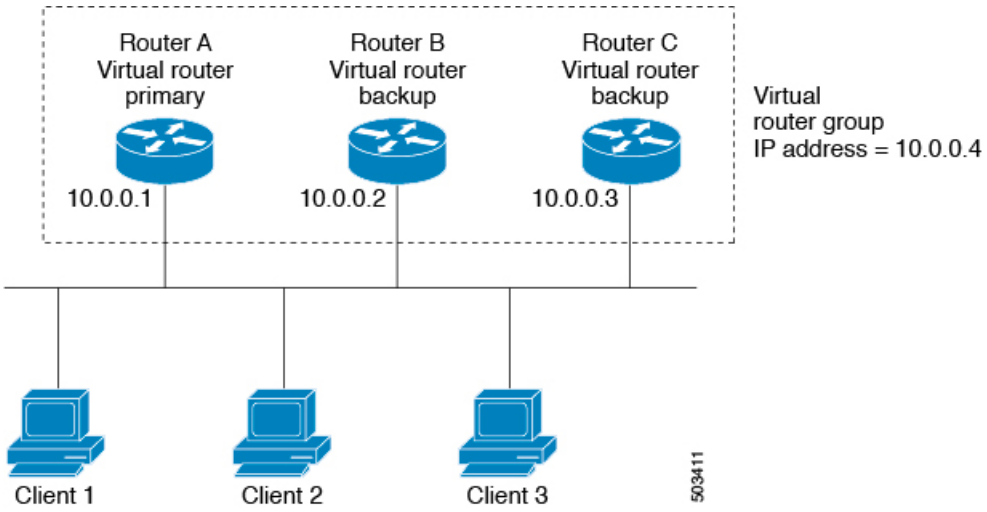
- Proxy ARP: The client uses ARP to resolve the destination, and a router responds with its MAC address.
- Routing protocol: The client listens to dynamic routing protocol updates (such as RIP) to build its routing table.
- ICMP Router Discovery Protocol (IRDP): The client runs an ICMP router discovery client.

When using VRRP, a group of routers share a virtual IP address. The primary router (IP address owner) forwards packets sent to this address, while backup routers monitor the primary's status. If the primary fails, the backup with the highest priority takes over the virtual IP address, ensuring uninterrupted service. When the original primary recovers, it resumes its role.



Note Packets received on a routed port destined for the VRRP virtual IP address terminate on the local router, regardless of whether it is the primary or a backup. These include ping and Telnet traffic. Packets received on a Layer 2 (VLAN) interface destined for the VRRP virtual IP address terminate on the primary router.

Figure 46: Basic VRRP Topology



Example: VRRP in a VLAN topology

In a basic VLAN topology, Routers A, B, and C form a VRRP group. The group's IP address matches the Ethernet interface address of Router A (10.0.0.1). Clients 1 through 3 are configured with 10.0.0.1 as their default gateway. Router A acts as the primary and forwards packets sent to the virtual IP address. If Router A fails, the backup router with the highest priority becomes the new primary and takes over the virtual IP address, maintaining network connectivity for the clients. When Router A recovers, it resumes its role as primary.

VRRP benefits

- Redundancy: Enables configuration of multiple routers as the default gateway, reducing 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, distributing the load more equitably.
- Multiple VRRP groups: Supports multiple VRRP groups on a router interface, enabling redundancy and load sharing in LAN topology.
- Multiple IP addresses: Allows management of multiple IP addresses, including secondary addresses, and supports VRRP configuration on each subnet.
- Preemption: Enables a higher priority backup router to preempt a backup that has taken over for a failing primary.
- Advertisement protocol: Uses a dedicated IANA standard multicast address (224.0.0.18) and protocol number 112 for VRRP advertisements, minimizing unnecessary multicasts and aiding in packet identification.
- VRRP tracking: Ensures the best VRRP router is primary by altering priorities based on interface states.

Multiple VRRP groups

Multiple VRRP groups refer to the configuration of more than one Virtual Router Redundancy Protocol (VRRP) group on a single physical router interface.

- Router interfaces can support multiple VRRP groups simultaneously.
- The number of supported VRRP groups depends on router processing and memory capabilities.
- An interface can act as a primary for one VRRP group and as a backup for one or more other groups.

Multiple VRRP groups can be configured on a single router interface, allowing for flexible redundancy and load sharing in network topologies.

- Router processing capability affects the number of VRRP groups supported.
- Router memory capability also impacts the number of supported groups.

In a topology with multiple VRRP groups, the interface can serve as a primary for one group and as a backup for others.

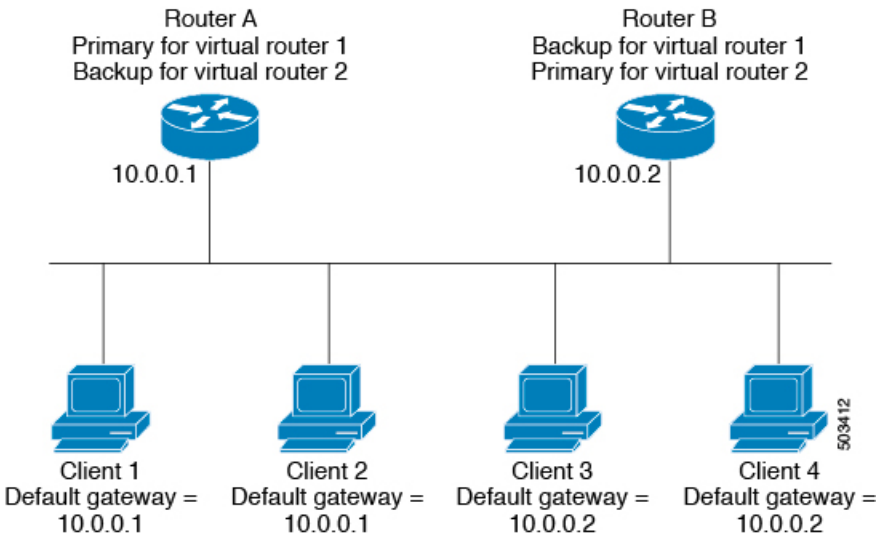
Table 43: VRRP Group Role Comparison

Group	Primary Router	Backup Router
VRRP group 1	Router A (IP 10.0.0.1)	Router B
VRRP group 2	Router B (IP 10.0.0.2)	Router A

Note For the number of supported VRRP groups, see the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide .

The following image 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 47: Load Sharing and Redundancy VRRP Topology



Example: Overlapping VRRP groups in a LAN 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

VRRP router priority is a key attribute in the VRRP redundancy scheme that determines the role of each router and the behavior during failover and preemption.

- The router that owns both the virtual IP address and the physical interface IP address functions as the primary, with a priority of 255.
- Backup routers are selected based on their configured priority; the router with the highest priority becomes the new primary if the current primary fails.
- Preemption allows a backup router with a higher priority to take over as primary, even if the current primary has not failed, unless preemption is disabled.

How VRRP router priority and preemption work

VRRP uses router priority to determine which router acts as primary and how failover occurs when the primary fails. Preemption controls whether a higher-priority backup can take over as primary.

- If the primary router fails, VRRP selects the backup router with the highest priority to become the new primary.
- If multiple backups have the same priority, the router with the higher IP address is selected as the new primary.

1. Primary router fails.
2. VRRP evaluates backup routers' priorities.
3. Backup with highest priority (or highest IP address if priorities are equal) becomes new primary.

- Priority 255: Assigned to the router owning the virtual and physical IP addresses (primary).
- Default priority: 100 (for backup routers unless configured otherwise).

Table 44: Priority Comparison Table

Router	Configured Priority	Selected as Primary?
Router A	255	Yes (initial primary)
Router B	101	Yes (if Router A fails and B has higher priority than C)
Router C	100 (default)	Yes (if B and C have same priority, higher IP address wins)

Note TIP: If preemption is disabled, a backup router with a higher priority will not take over as primary unless the current primary fails or recovers.

VRRP priority and preemption in action

For example, if Router A (primary) fails, VRRP selects Router B (priority 101) over Router C (priority 100) as the new primary. If both backups have the same priority, the router with the higher IP address becomes primary. If preemption is enabled and Router C comes online with a higher priority than the current primary, VRRP selects Router C as the new primary, even if the current primary has not failed.

vPCs and VRRP

vPCs and VRRP are technologies that work together to provide high availability and redundancy in Cisco Nexus 9000 Series switches.

- vPCs allow links physically connected to two different switches to appear as a single port channel to a third device.
- VRRP provides router redundancy by designating primary and backup routers.
- vPCs forward traffic through both the primary and backup VRRP routers.

vPCs and VRRP are used together to ensure continuous network connectivity and redundancy in Cisco Nexus 9000 Series switches.

Note You should configure VRRP on the primary vPC peer device as active and VRRP on the vPC secondary device as standby.

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

For details on configuring VRRP priority, see the Configuring VRRP Priority section.

VRRP advertisements

VRRP advertisements are periodic messages sent by the VRRP primary router to other VRRP routers in the same group to communicate its priority and state.

- The VRRP primary sends advertisements to other routers in the same group.
- Advertisements communicate the priority and state of the primary.
- Advertisements are encapsulated in IP packets and sent to the IP multicast address assigned to the VRRP group.

The VRRP primary sends advertisements once every second by default, but you can configure a different advertisement interval.

VRRP authentication

- VRRP supports two authentication functions: no authentication and plain text authentication.
- Authentication ensures that only valid VRRP packets are accepted by the router.
- Packets are rejected if authentication schemes or text strings differ between the router and incoming packets.

VRRP authentication methods and packet rejection criteria

VRRP provides two authentication options and enforces strict packet validation based on authentication configuration.

- No authentication
- Plain text authentication

VRRP rejects packets in 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.

VRRP tracking

VRRP tracking is a mechanism that enables a VRRP router to monitor the state of interfaces or configured objects and adjust its priority in a VRRP group accordingly.

- Tracks the state of an interface or a configured object.
- Adjusts the VRRP router's priority based on the tracked state.
- Restores the original priority when the tracked state returns to up.

VRRP tracking options

VRRP supports the following options for tracking:

- Native interface tracking—Tracks the state of an interface and uses that state to determine the priority of the VRRP router in a VRRP group. The tracked state is down if the interface is down or if the interface does not have a primary IP address.
- Object tracking—Tracks the state of a configured object and uses that state to determine the priority of the VRRP router in a VRRP group. See Configuring Object Tracking for more information on object tracking.

If the tracked state (interface or object) goes down, VRRP updates the priority based on what you configure the new priority to be for the tracked state. When the tracked state comes up, VRRP restores the original priority for the virtual router group.



Note VRRP does not support Layer 2 interface tracking.

VRRP tracking in use

For example, you might want to lower the priority of a VRRP group member if its uplink to the network goes down so another group member can take over as primary for the VRRP group. See the Configuring VRRP Interface State Tracking section for more information.

BFD for VRRP

BFD for VRRP is a protocol integration that enables rapid detection of forwarding and path failures between two adjacent devices in a VRRP environment.

- Provides subsecond failure detection between two adjacent devices.
- Can be less CPU-intensive than protocol hello messages.
- Some BFD load can be distributed onto the data plane on supported modules.

BFD (Bidirectional Forwarding Detection) is a detection protocol that provides fast-forwarding and path-failure detection times. For more information, see the Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide .

Information about VRRPv3 and VRRS

- **VRRPv3** enables a group of switches to form a single virtual switch, providing redundancy and reducing single points of failure in a network.
- **VRRS** improves the scalability of VRRPv3 by providing stateless redundancy services to VRRS pathways and clients, monitoring VRRPv3 status and distributing it to registered clients.
- VRRS pathways and clients use VRRPv3 state information to alter their behavior, enabling scalable first-hop gateway redundancy and supporting both stateless and stateful failovers.

VRRPv3 allows multiple switches to act as a single virtual switch (VRRPv3 group), which serves as the default gateway for LAN clients. VRRS extends this by monitoring VRRPv3 and distributing its state to VRRS pathways and clients, enabling scalable and resilient network configurations.

VRRS pathways and clients in action

When a VRRPv3 group changes state, VRRS pathways and clients respond by performing actions such as shutting down interfaces or updating accounting logs, depending on the received state. For example, a VRRS pathway can configure a virtual address across hundreds of interfaces, and its virtual gateway state follows the state of the FHRP VRRS server.

VRRPv3 benefits

- Interoperability in multi-vendor environments
- Support for the IPv4 and IPv6 address families
- Improved scalability through the use of VRRS pathways

VRRPv3 object tracking

VRRPv3 object tracking is a feature that enables a VRRPv3 router to monitor the state of a configured object and adjust its priority in a VRRPv3 group accordingly.

- Tracks the state of a configured object.
- Adjusts the VRRPv3 router's priority based on the tracked object's state.
- Priority is decremented or incremented by a configured value when the tracked object goes down or comes up, respectively.

Beginning with Cisco NX-OS Release 9.2(2), VRRPv3 supports object tracking, which tracks the state of a configured object and uses that state to determine the priority of the VRRPv3 router in a VRRPv3 group.

- If the tracked object goes down, VRRPv3 decrements the priority by the configured value (default is 10).
- If the same tracked object goes down again, no further action is taken.
- When the tracked object comes up, VRRPv3 increments the priority by the configured value.

Note VRRPv3 does not support Layer 2 interface tracking or native interface tracking.

High availability

- High availability is achieved in VRRP by supporting stateful restarts and stateful switchovers.
- A stateful restart occurs when the VRRP process fails and is restarted.
- A stateful switchover occurs when the active supervisor switches to the standby supervisor.
- After a switchover, the system applies the run-time configuration.
- VRRPv3 does not support stateful switchovers.

Virtualization support

Virtualization support refers to the capability of VRRP to operate within virtual routing and forwarding (VRF) instances.

- Enables VRRP to function in environments with multiple VRFs.
- Allows separation of routing tables for different network segments.

Guidelines and limitations for VRRP

- VRRP cannot be configured on the management interface.
- When VRRP is enabled, you should replicate the VRRP configuration across devices in your network.
- Do not configure more than one first-hop redundancy protocol on the same interface.
- You must configure an IP address for the interface on which you configure VRRP and enable that interface before VRRP becomes active.
- All Layer 3 configurations on an interface are removed when you change the interface VRF membership, the 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.
- BFD for VRRP can only be configured between two routers.

Guidelines and limitations for VRRPv3

- VRRPv3 is supported on specific Cisco Nexus switches and interfaces, with a maximum of 4095 VRRPv3 groups and VRRS pathways on Cisco Nexus 9504, 9508, and 9516 switches with -R line cards (Release 9.3(1)).
- VRRPv3 is designed for use over multi-access, multicast, or broadcast-capable Ethernet LANs and is not intended as a replacement for existing dynamic protocols.
- VRRPv3 is supported only on Ethernet, Fast Ethernet, Gigabit Ethernet interfaces, bridge group virtual interfaces (BVIs), and VLANs.
- When VRRPv3 is in use, VRRPv2 is unavailable; you must disable any VRRPv2 configuration before configuring VRRPv3.
- VRRS is currently available only for use with VRRPv3.
- Use VRRPv3 millisecond timers only where necessary and with careful consideration and testing; millisecond values work only under favorable circumstances and are compatible with third-party vendors that support VRRPv3.
- Full network redundancy requires VRRPv3 to operate over the same network path as the VRRS pathway redundant interfaces, with the following restrictions:
  - VRRS pathways should use the same physical interface as the parent VRRPv3 group or be configured on a subinterface with the same physical interface as the parent VRRPv3 group.
  - VRRS pathways can be configured on switch virtual interfaces (SVIs) only if the associated VLAN shares the same trunk as the VLAN on which the parent VRRPv3 group is configured.
- Unlike VRRPv2, VRRPv3 does not support bidirectional forwarding for faster failure detection and native interface tracking.
- Beginning with Cisco NX-OS Release 10.6(2)F, VRRP is supported on Cisco Nexus N9324C-SE1U and N9348Y2C6D-SE1U platform switches.

Guidelines and limitations of VRRPv3 object tracking

The following guidelines and limitations apply to VRRPv3 object tracking:

- Beginning with Cisco NX-OS Release 9.2(2), all Cisco Nexus 9000 Series switches and line cards support VRRPv3 object tracking.
- It is recommended not to use VRRPv3 object tracking in a vPC domain.
- You must create the object before configuring object tracking.

Default settings for VRRP parameters

This topic lists the default settings for VRRP parameters.

Table 45: Default VRRP parameters

Parameters	Default
VRRP	Disabled
Advertisement interval	1 second
Authentication	No authentication
Preemption	Enabled
Priority	100

Default settings for VRRPv3 parameters

The following table lists the default settings for VRRPv3 parameters.

Table 46: Default VRRPv3 parameters

Parameters	Default
VRRPv3	Disabled
VRRS	Disabled
VRRPv3 secondary address matching	Enabled
Priority of a VRRPv3 group	100
VRRPv3 advertisement timer	1000 milliseconds

Configure VRRP

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

Enable VRRP

You must globally enable VRRP before you configure and enable any VRRP groups.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 [**no**] **feature vrrp**

Example:

```
switch(config)# feature vrrp
```

Enables VRRP. Use the **no** form of this command to disable VRRP.

Step 3 (Optional) **copy running-config startup-config**

Example:

```
switch(config)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

Configure a VRRP group

Use this task to configure 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 intended only 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 address. 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 have configured an IP address on the interface. See Configure IPv4 address, on page 35 .

Procedure

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 **vrrp** *number*

Example:

```
switch(config-if)# vrrp 250
switch(config-if-vrrp)#
```

Creates a virtual router group. The range is 1–255.

Step 4 **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 5 **no shutdown**

Example:

```
switch(config-if-vrrp)# no shutdown
```

Enables the VRRP group, which is disabled by default.

Step 6 (Optional) **show vrrp**

Example:

```
switch(config-if-vrrp)# show vrrp
```

Displays a summary of VRRP information.

Step 7 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if-vrrp)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

Configure VRRP priority

The valid priority range for a virtual router is from 1 to 254 (1 is the lowest priority and 254 is the highest). The default priority value for backups is 100. For devices whose interface IP address is the same as the primary virtual IP address (the primary), the default value is 255.

If you configure VRRP on a vPC-enabled interface, you can optionally configure the upper and lower threshold values to control when to fail over to the vPC trunk. If the backup router priority falls below the lower threshold, VRRP sends all backup router traffic across the vPC trunk to forward through the primary VRRP router. VRRP maintains this scenario until the backup VRRP router priority increases above the upper threshold.

Before you begin

Ensure that you have configured an IP address on the interface. See Configure IPv4 address, on page 35 .

Ensure that you have enabled VRRP. (see the Configuring VRRP section).

Procedure

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 **vrrp** *number*

Example:

```
switch(config-if)# vrrp 250
switch(config-if-vrrp)#
```

Creates a virtual router group.

Step 4 **shutdown**

Example:

```
switch(config-if-vrrp)# shutdown
```

Disables the VRRP group.

Step 5 **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 a 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.

Optionally, sets the upper and lower threshold values that are used by vPC to determine when to fail over to the vPC trunk. The *lower-value* range is 1–255. The default is 1. The *upper-value* range is 1–255. The default is 255.

Step 6 **no shutdown**

Example:

```
switch(config-if-vrrp)# no shutdown
```

Enables the VRRP group.

Step 7 (Optional) **show vrrp**

Example:

```
switch(config-if-vrrp)# show vrrp
```

Displays a summary of VRRP information.

Step 8 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if-vrrp)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

Configure VRRP authentication

You can configure simple text authentication for a VRRP group.

Before you begin

Ensure that you have configured an IP address on the interface (see Configure IPv4 address, on page 35).

Ensure that you have enabled VRRP (see the Configuring VRRP section).

Ensure that the authentication configuration is identical for all VRRP devices in the network.

Procedure

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 **vrrp** *number*

Example:

```
switch(config-if)# vrrp 250
switch(config-if-vrrp)#
```

Creates a virtual router group.

Step 4 **shutdown**

Example:

```
switch(config-if-vrrp)# shutdown
```

Disables the VRRP group.

Step 5 **authentication text** *password*

Example:

```
switch(config-if-vrrp)# authentication text aPassword
```

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. The text password is up to eight alphanumeric characters.

Step 6 **no shutdown**

Example:

```
switch(config-if-vrrp)# no shutdown
```

Enables the VRRP group, which is disabled by default.

Step 7 (Optional) **show vrrp**

Example:

```
switch(config-if-vrrp)# show vrrp
```

Displays a summary of VRRP information.

Step 8 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if-vrrp)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

Configure time intervals for advertisement packets

Configure the time intervals for advertisement packets to control how often VRRP advertisements are sent.

You can configure the time intervals for advertisement packets.

Before you begin

Ensure that you have configured an IP address on the interface (see Configure IPv4 address, on page 35 .

Ensure that you have enabled VRRP (see the Configuring VRRP section).

Procedure

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 **vrrp** *number*

Example:

```
switch(config-if)# vrrp 250
switch(config-if-vrrp)#
```

Creates a virtual router group.

Step 4 **shutdown**

Example:

```
switch(config-if-vrrp)# shutdown
```

Disables the VRRP group.

Step 5 **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 255. The default is 1 second.

Step 6 **no shutdown**

Example:

```
switch(config-if-vrrp)# no shutdown
```

Enables the VRRP group.

Step 7 (Optional) **show vrrp**

Example:

```
switch(config-if-vrrp)# show vrrp
```

Displays a summary of VRRP information.

Step 8 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if-vrrp)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

Disable preemption

You can disable preemption for a VRRP group member. If you disable preemption, a higher-priority backup router does not take over for a lower-priority primary router. Preemption is enabled by default.

Before you begin

Ensure that you have configured an IP address on the interface. See Configure IPv4 address, on page 35 .

Ensure that you have enabled VRRP. See the Configuring VRRP section.

Procedure

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 **vrrp** *number*

Example:

```
switch(config-if)# vrrp 250
switch(config-if-vrrp)#
```

Creates a virtual router group.

Step 4 **shutdown**

Example:

```
switch(config-if-vrrp)# shutdown
```

Disables the VRRP group.

Step 5 **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 6 **no shutdown**

Example:

```
switch(config-if-vrrp)# no shutdown
```

Enables the VRRP group.

Step 7 (Optional) **show vrrp**

Example:

```
switch(config-if-vrrp)# show vrrp
```

Displays a summary of VRRP information.

Step 8 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if-vrrp)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

Configure VRRP interface state tracking

Use this task to configure VRRP interface state tracking, which allows the virtual router's priority to change dynamically based on the state of a tracked interface.

Interface state tracking changes the priority of the virtual router based on the state of another interface in the device. 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 VRRP does not support Layer 2 interface tracking.

Before you begin

Ensure that you have configured an IP address on the interface (see Configure IPv4 address, on page 35).

Ensure that you have enabled VRRP (see the Configuring VRRP section).

Ensure that you have enabled the virtual router (see the Configuring VRRP Groups section).

Ensure that you have enabled preemption on the interface.

Procedure

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 **vrrp** *number*

Example:

```
switch(config-if)# vrrp 250
      switch(config-if-vrrp)#
```

Creates a virtual router group.

Step 4 **shutdown**

Example:

```
switch(config-if-vrrp)# shutdown
```

Disables the VRRP group.

Step 5 **track interface** *type slot/port* **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 6 **no shutdown**

Example:

```
switch(config-if-vrrp)# no shutdown
```

Enables the VRRP group.

Step 7 (Optional) **show vrrp**

Example:

```
switch(config-if-vrrp)# show vrrp
```

Displays a summary of VRRP information.

Step 8 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if-vrrp)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

Configure VRRP object tracking

Use this task to configure VRRP object tracking on your device.

You can track an IPv4 object using VRRP.

Before you begin

Make sure that VRRP is enabled.

Configure object tracking using the commands in Configuring Object Tracking section.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **interface type number**

Example:

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

Specifies an interface and enters interface configuration mode.

Step 3 **vrrp** *number* **address-family ipv4**

Example:

```
switch(config-if)# vrrp 5
                        address-family ipv4
                        switch(config-if-vrrp-group)#
```

Creates a VRRP group for IPv4 and enters VRRP vrrp number address-family ipv4 group configuration mode. The range is from 1 to 255.

Step 4 **track** *object-number* **decrement** *number*

Example:

```
switch(config-if-vrrp-group)# track 1
                        decrement 2
```

Creates a virtual router group. The range is from 1 to 255.

Step 5 (Optional) **show running-config vrrp**

Example:

```
switch(config-if-vrrp-group)# show
                        running-config vrrp
```

Displays the running configuration for VRRP.

Step 6 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if-vrrp-group)# copy
                        running-config startup-config
```

Saves this configuration change.

Configure VRRPv3

Enable VRRPv3 and VRRS

You must globally enable VRRPv3 before you can configure and enable any VRRPv3 groups.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 [**no**] **feature vrrpv3**

Example:

```
switch(config)# feature vrrpv3
```

Enables VRRP version 3 and Virtual Router Redundancy Service (VRRS). The **no** form of this command disables VRRPv3 and VRRS.

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.

Step 3 (Optional) **copy running-config startup-config**

Example:

```
switch(config)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

Create VRRPv3 groups

You can create a VRRPv3 group, assign the virtual IP address, and enable the group.

Before you begin

Make sure that VRRPv3 is enabled.

Make sure that you have configured an IP address on the interface.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **interface ethernet** *slot/port*

Example:

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

Enters interface configuration mode.

Step 3 **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.

Step 4 (Optional) **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.

To utilize secondary IP addresses in a VRRPv3 group, you must first configure a primary IP address on the same group.

Step 5 (Optional) **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.

Step 6 (Optional) **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) **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) **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) **timers advertise** *interval*

Example:

```
switch(config-if-vrrpv3-group)# timers advertise 1000
```

Sets the advertisement timer in milliseconds. The range is 100–40950.

Cisco recommends that you set this timer to a value greater than or equal to 1 second.

Step 10 (Optional) **vrrp2**

Example:

```
switch(config-if-vrrpv3-group)# vrrp2
```

Enables support for VRRPv2 simultaneously to ensure interoperability with devices that support only VRRPv2.

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) **vrrs leader** *vrrs-leader-name*

Example:

```
switch(config-if-vrrpv3-group)# vrrs leader leader1
```

Specifies a leader's name to be registered with VRRS.

Step 12 (Optional) **shutdown**

Example:

```
switch(config-if-vrrpv3-group)# shutdown
```

Disables the VRRP configuration for the VRRPv3 group.

Step 13 (Optional) **show fhrp** [*interface-type interface-number*] [**verbose**]

Example:

```
switch(config-if-vrrpv3-group)# show fhrp ethernet 2/1 verbose
```

Displays First Hop Redundancy Protocol (FHRP) information. Use the **verbose** keyword to view detailed information.

Step 14 (Optional) **show vrrpv3** *interface-type interface-number*

Example:

```
switch(config-if-vrrpv3-group)# show vrrpv3 ethernet 2/1
```

Displays the VRRPv3 configuration information for the specified interface.

Step 15 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if-vrrpv3-group)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

Configure VRRPv3 control groups

You can configure VRRPv3 control groups.

Before you begin

Make sure that VRRPv3 is enabled.

Make sure that you have configured an IP address on the interface.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **interface ethernet** *slot/port*

Example:

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

Enters interface configuration mode.

Step 3 **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.

You can use the **secondary** keyword to configure additional IP addresses on the interface.

Step 4 **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) **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) **shutdown**

Example:

```
switch(config-if-vrrpv3-group)# shutdown
```

Disables the VRRP configuration for the VRRPv3 group.

Step 7 (Optional) **show fhrp** [*interface-type interface-number*] [**verbose**]

Example:

```
switch(config-if-vrrpv3-group)# show fhrp ethernet 2/1 verbose
```

Displays First Hop Redundancy Protocol (FHRP) information. Use the **verbose** keyword to view detailed information.

Step 8 (Optional) **show vrrpv3** *interface-type interface-number*

Example:

```
switch(config-if-vrrpv3-group)# show vrrpv3 ethernet 2/1
```

Displays the VRRPv3 configuration information for the specified interface.

Step 9 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if-vrrpv3-group)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

Configure VRRPv3 object tracking

Use this task to configure object tracking with VRRPv3, allowing the system to monitor the state of IPv4 or IPv6 objects and adjust the VRRPv3 group priority if the tracked object state changes.

You can track an IPv4 or IPv6 object using VRRPv3.

Before you begin

Make sure that VRRPv3 is enabled.

Configure object tracking using the commands in Configuring Object Tracking section.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **interface type number**

Example:

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

Specifies an interface and enters interface configuration mode.

Step 3 **vrrpv3** *number* **address-family [ipv4 | ipv6]**

Example:

```
switch(config-if)# vrrpv3 5
                        address-family ipv6
                        switch(config-if-vrrpv3-group)#
```

Creates a VRRPv3 group for IPv4 or IPv6 and enters VRRPv3 group configuration mode. The range is from 1 to 255.

Step 4 **track** *object-number* **decrement** *number*

Example:

```
switch(config-if-vrrpv3-group)# object-track 1
                        decrement 2
```

Configures the process to track the state of the IPv4 or IPv6 object using the VRRPv3 group. VRRPv3 on the interface registers with the tracking process to be informed of any changes to the object in the VRRPv3 group. If the object state on the interface goes down, the priority of the VRRPv3 group is reduced by the decrement number specified.

Step 5 (Optional) **show running-config vrrpv3**

Example:

```
switch(config-if-vrrp-group)# show
                        running-config vrrp
```

Displays the running configuration for VRRPv3.

Step 6 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if-vrrp-group)# copy
                        running-config startup-config
```

Saves this configuration change.

Configure VRRS pathways

You can configure a Virtual Router Redundancy Service (VRRS) pathway. In scaled environments, VRRS pathways should be used in combination with VRRPv3 control groups.

Before you begin

Make sure that VRRPv3 is enabled.

Make sure that you have configured an IP address on the interface.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **interface ethernet** *slot/port*

Example:

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

Enters interface configuration mode.

Step 3 **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.

You can use the **secondary** keyword to configure additional IP addresses on the interface.

Step 4 **vrrs pathway** *vrrs-tag*

Example:

```
switch(config-if)# vrrs pathway path1
switch(config-if-vrrs-pw)#
```

Defines the VRRS pathway for a VRRS group and enters VRRS pathway configuration mode.

The *vrrs-tag* argument specifies the name of the VRRS tag that is being associated with the pathway.

Step 5 **mac address** { *mac-address* | **inherit** }

Example:

```
switch(config-if-vrrs-pw)# mac address fe24.fe24.fe24
```

Specifies a MAC address for the pathway.

The **inherit** keyword causes the pathway to inherit the virtual MAC address of the VRRPv3 group with which the pathway is associated.

Step 6 **address** *ip-address*

Example:

```
switch(config-if-vrrs-pw)# address 209.165.201.10
```

Defines the virtual IPv4 or IPv6 address for a pathway.

A VRRPv3 group is capable of controlling more than one pathway.

Step 7 (Optional) **show vrrs pathway** *interface-type interface-number*

Example:

```
switch(config-if-vrrs-pw)# show vrrs pathway ethernet 1/2
```

Displays the VRRS pathway information for different pathway states, such as active, inactive, and not ready.

Step 8 (Optional) **copy running-config startup-config**

Example:

```
switch(config-if-vrrs-pw)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

Verify the VRRP configuration

To display VRRP configuration information, perform one of the following tasks.

Command	Purpose
show interface *interface-type*	Displays the virtual router configuration for an interface.
show fhrp *interface-type interface-number*	Displays First Hop Redundancy Protocol (FHRP) information.
show vrrp [*group-number*]	Displays the VRRP status for all groups or for a specific VRRP group.

Verify the VRRPv3 configuration

To display VRRPv3 configuration information, perform one of the following tasks:

Command	Purpose
show vrrpv3 [**all** | **brief** | **detail**]	Displays the VRRPv3 configuration information.
show vrrpv3 *interface-type interface-number*	Displays the VRRPv3 configuration information for a specific interface.
show vrrs client [*client-name*]	Displays the VRRS client information.
show vrrs pathway [*interface-type interface-number*]	Displays the VRRS pathway information for different pathway states, such as active, inactive, and not ready.
show vrrs server	Displays the VRRS server information.
show vrrs tag [*tag-name*]	Displays the VRRS tag information.

Monitor and clear VRRP statistics

To display VRRP statistics, use the following commands.

Command	Purpose
show vrrp statistics	Displays the VRRP statistics.

Use the **clear vrrp statistics** command to clear the VRRP statistics for all interfaces on the device.

Monitor and clear VRRPv3 statistics

To display VRRPv3 statistics, use the following commands.

Command	Purpose
show vrrpv3 statistics	Displays the VRRPv3 statistics.

Use the **clear vrrpv3 statistics** command to clear the VRRPv3 statistics for all interfaces on the device.

Configuration examples for VRRP

This topic provides configuration examples for VRRP groups, including group properties and sample configurations for Router A and Router B.

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.
  - Pre-emption is enabled.
- Group 5:
  - Router B becomes the primary for this group with priority 200.
  - Advertising interval is 30 seconds.
  - Pre-emption 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 of 1 second.

- Pre-emption is disabled.

Router A

```
switch (config)# <!--Modified interface ethernet as per CSCwc29303
(Sneha)--><userinput>interface ethernet 1/1</userinput>
switch (config-if)# <!--Modified ip address as per CSCwc29303 (Sneha)--><userinput>ip address
 10.1.0.1/16</userinput>
switch (config-if)# <userinput>no shutdown</userinput>
switch (config-if)# <userinput>vrrp 1</userinput>
switch (config-if-vrrp)# <userinput>priority 120</userinput>
switch (config-if-vrrp)# <userinput>authentication text cisco</userinput>
switch (config-if-vrrp)# <userinput>advertisement-interval 3</userinput>
switch (config-if-vrrp)# <userinput>address 10.1.0.10</userinput>
switch (config-if-vrrp)# <userinput>no shutdown</userinput>
switch (config-if-vrrp)# <userinput>exit</userinput>
switch (config-if)# <userinput>vrrp 5</userinput>
switch (config-if-vrrp)# <userinput>priority 100</userinput>
switch (config-if-vrrp)# <userinput>advertisement-interval 30</userinput>
switch (config-if-vrrp)# <userinput>address 10.1.0.50</userinput>
switch (config-if-vrrp)# <userinput>no shutdown</userinput>
switch (config-if-vrrp)# <userinput>exit</userinput>
switch (config-if)# <userinput>vrrp 100</userinput>
switch (config-if-vrrp)# <userinput>no preempt</userinput>
switch (config-if-vrrp)# <userinput>address 10.1.0.100</userinput>
switch (config-if-vrrp)# <userinput>no shutdown</userinput>
```

Router B

```
switch (config)# <!--Modified interface ethernet as per CSCwc29303
(Sneha)--><userinput>interface ethernet 1/1</userinput>
switch (config-if)# <!--Modified ip address as per CSCwc29303 (Sneha)--><userinput>ip address
 10.1.0.2/16</userinput>
switch (config-if)# <userinput>no shutdown</userinput>
switch (config-if)# <userinput>vrrp 1</userinput>
switch (config-if-vrrp)# <userinput>priority 100</userinput>
switch (config-if-vrrp)# <userinput>authentication text cisco</userinput>
switch (config-if-vrrp)# <userinput>advertisement-interval 3</userinput>
switch (config-if-vrrp)# <!--Modified vrrp address as per CSCwc29303
(Sneha)--><userinput>address 10.1.0.10</userinput>
switch (config-if-vrrp)# <userinput>no shutdown</userinput>
switch (config-if-vrrp)# <userinput>exit</userinput>
switch (config-if)# <userinput>vrrp 5</userinput>
switch (config-if-vrrp)# <userinput>priority 200</userinput>
switch (config-if-vrrp)# <userinput>advertisement-interval 30</userinput>
switch (config-if-vrrp)# <userinput>address 10.2.0.50</userinput>
switch (config-if-vrrp)# <userinput>no shutdown</userinput>
switch (config-if-vrrp)# <userinput>exit</userinput>
switch (config-if)# <userinput>vrrp 100</userinput>
switch (config-if-vrrp)# <userinput>no preempt</userinput>
switch (config-if-vrrp)# <userinput>address 10.2.0.100</userinput>
switch (config-if-vrrp)# <userinput>no shutdown</userinput>
```

Configuration examples for VRRPv3

This topic provides configuration examples for enabling and customizing VRRPv3, configuring VRRPv3 control groups, object tracking, and VRRS pathways.

This example shows how to enable VRRPv3 and create and customize a VRRPv3 group:

```
switch# configure terminal
switch(config)# feature vrrpv3
switch(config)# interface ethernet 4/6
switch(config-if)# vrrpv3 5 address-family ipv4
switch(config-if-vrrp3-group)# address 209.165.200.225 primary
switch(config-if-vrrp3-group)# description group3
switch(config-if-vrrp3-group)# match-address
switch(config-if-vrrp3-group)# preempt delay minimum 30
switch(config-if-vrrpv3-group)# show fhrp ethernet 4/6 verbose
switch(config-if-vrrpv3-group)# show vrrpv3 ethernet 4/6
```

This example shows how to configure a VRRPv3 control group:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ip address 209.165.200.230 255.255.255.224
switch(config-if)# vrrpv3 5 address-family ipv4
switch(config-if-vrrpv3-group)# address 209.165.200.227 primary
switch(config-if-vrrpv3-group)# vrrs leader leader1
switch(config-if-vrrpv3-group)# shutdown
switch(config-if-vrrpv3-group)# show fhrp ethernet 1/2 verbose
switch(config-if-vrrpv3-group)# show vrrpv3 ethernet 1/2
```

This example shows how to configure object tracking for VRRPv3:

```
track 1 interface Ethernet1/12 ip routing
track 2 interface Ethernet1/12 ipv6 routing
track 3 interface Ethernet1/12 line-protocol
track 4 interface Ethernet1/12.1 ip routing
track 5 interface Ethernet1/12.1 ipv6 routing
track 6 interface Ethernet1/12.1 line-protocol
track 7 interface loopback1 ip routing
track 8 interface loopback1 ipv6 routing
track 9 interface loopback1 line-protocol
track 10 interface port-channel1 ip routing
track 11 interface port-channel1 ipv6 routing
track 12 interface port-channel1 line-protocol
track 13 ip route 170.10.10.10/24 reachability
track 14 ip route 180.10.10.0/24 reachability hmm
track 15 ipv6 route 2001::170:10:10:10/128 reachability
track 16 list boolean and
object 1
object 2
interface Vlan10
vrrpv3 10 address-family ipv4
timers advertise 100
priority 200
object-track 1 decrement 2
object-track 2 decrement 2
object-track 3 decrement 2
object-track 4 decrement 2
object-track 5 decrement 2
object-track 6 decrement 2
object-track 7 decrement 2
object-track 8 decrement 2
object-track 9 decrement 2
object-track 10 decrement 2
address 10.10.10.3 primary
interface Vlan10
vrrpv3 10 address-family ipv6
timers advertise 100
```

```
priority 200
object-track 1 decrement 4
object-track 2 decrement 4
object-track 3 decrement 4
object-track 4 decrement 4
object-track 5 decrement 4
object-track 6 decrement 4
object-track 7 decrement 4
object-track 8 decrement 4
```

This example shows how to configure VRRS pathways:

```
switch#
configure terminal
switch(config)#
interface ethernet 1/2
switch(config-if)#
ip address 209.165.200.230 255.255.255.224
switch(config-if)#
vrrs pathway path1
switch(config-if-vrrs-pw)#
mac address inherit
switch(config-if-vrrs-pw)#
address 209.165.201.10
switch(config-if-vrrs-pw)#
show vrrs pathway ethernet 1/2
```

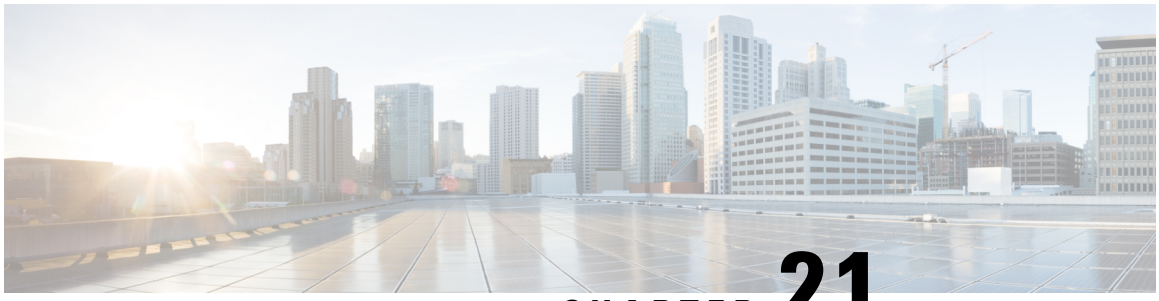
Additional references

Related documents for VRRP

This section lists the related documents for VRRP.

Table 47: Related documents for VRRP

Related Topic	Document Title
Configuring the Hot Standby Routing Protocol (HSRP)	Configuring HSRP, on page 609
Configuring high availability	Cisco Nexus 9000 Series NX-OS High Availability and Redundancy Guide



CHAPTER 21

Configuring Object Tracking

This chapter contains the following sections:

- Information about object tracking, on page 677
- Configuration examples for object tracking, on page 679
- Guidelines and limitations for object tracking, on page 680
- Default settings, on page 680
- Configure object tracking, on page 681
- Verify the object tracking configuration, on page 691
- Configuration examples for object tracking, on page 691
- Related topics, on page 692
- Related documents, on page 692

Information about object tracking

Object tracking is a feature that allows you to monitor specific objects on a device and take action when their state changes.

- Tracks objects such as interface line protocol state, IP routing, and route reachability.
- Enables actions to be triggered when the tracked object's state changes.
- Increases network availability and shortens recovery time if an object state goes down.

Object tracking overview

Object tracking is a feature that enables monitoring of specific objects on a device and allows actions to be taken when the state of those objects changes.

- Tracks objects such as interface line protocol state, IP routing state, and route reachability.
- Allows multiple clients to register interest and take action when a tracked object changes state.
- Each tracked object is identified by a unique number for client configuration.

Clients and object types for object tracking

The object tracking feature allows multiple clients to use tracked objects to modify their behavior when the object state changes. Clients register with the tracking process and can take different actions based on the tracked object's state.

Clients that use object tracking include:

- Embedded Event Manager (EEM)
- Hot Standby Redundancy Protocol (HSRP)
- Virtual port channel (vPC)
- Virtual Router Redundancy Protocol (VRRP) and VRRPv3

The following object types can be tracked:

- 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 or IPv6 address and if IPv4 or IPv6 routing is enabled and active.
- IP route reachability—Tracks whether an IPv4 or IPv6 route exists and is reachable from the local device.

Example of object tracking with HSRP

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 and cause a switchover to a backup router that has better network connectivity.

Object track list

An object track list is a mechanism that enables tracking of the combined states of multiple objects using Boolean logic or threshold-based evaluation.

- Supports Boolean "and" function: All objects in the track list must be up for the track list to be up.
- Supports Boolean "or" function: At least one object in the track list must be up for the track list to be up.
- Supports threshold percentage and threshold weight evaluation for determining the up or down state of the track list.

Object track list capabilities and usage

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 exceed the track list weight up threshold, the track list is in an up state. If the combined weights of all the down objects exceed the track list weight down threshold, the track list is in the down state.

Other entities, such as virtual port channels (vPCs), can use an object track list to modify the state of a vPC based on the state of the multiple peer links that create the vPC.

See the Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide for more information on vPCs.

See the Configuring an Object Track List with a Boolean Expression section for more information on track lists.

High availability

High availability is a network feature that ensures continuous operation by minimizing downtime through mechanisms such as stateful restarts and switchovers.

- Object tracking supports high availability through stateful restarts when the tracking process crashes.
- Object tracking supports stateful switchover on dual-supervisor systems.
- Object tracking can modify client behavior to improve overall network availability.

Object tracking enables high availability by maintaining configuration and state during process restarts and supervisor switchovers.

Virtualization support

Virtualization support in object tracking enables monitoring of route reachability for objects within different VRF instances.

- By default, Cisco NX-OS tracks the route reachability state of objects in the default VRF.
- You can configure object tracking for nondefault VRFs by assigning the object to the desired VRF.

Object tracking supports both default and nondefault VRF instances. To track objects in a nondefault VRF, configure the object as a member of that VRF.

For more information, see the Configuring Object Tracking for a Nondefault VRF section.

Configuration examples for object tracking

Object tracking configuration allows you to monitor the reachability of a route and associate the tracking with a specific VRF instance.

- Enables tracking of route reachability.
- Associates the tracked object with a VRF (Virtual Routing and Forwarding) instance.

- Configuration is saved to persist across reloads.

Object tracking configuration for route reachability using VRF

This section provides a configuration example for tracking the reachability of a route and associating it with a VRF instance.

- Configure object tracking for a specific route.
- Assign the tracked object to a VRF.
- Save the configuration.

1. Enter global configuration mode.
2. Configure object tracking for the route 209.165.201.0/8.
3. Associate the tracked object with VRF Red.
4. Save the running configuration to startup configuration.

Example: Configuring object tracking for route reachability with VRF Red

The following example shows the CLI commands to configure object tracking for route 209.165.201.0/8 and associate it with VRF Red:

```
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
```

Guidelines and limitations for object tracking

- Supports Ethernet, subinterfaces, port channels, loopback interfaces, and VLAN interfaces.
- Supports one tracked object per HSRP group.
- VRRP and VRRPv3 support object tracking. For more information and configuration instructions, see Configuring VRRP .

Default settings

The following table lists the default settings for object tracking parameters.

Table 48: Default object tracking parameters

Parameters	Default
Tracked object VRF	Member of default VRF

Configure object tracking

Configuring object tracking refers to the process of setting up mechanisms to monitor and track the state of network objects, such as IP SLA operations.

For detailed configuration steps and additional information, refer to the following guide:

- Cisco Nexus 9000 Series NX-OS IP SLAs Configuration Guide

Configure object tracking for an interface

You can configure Cisco NX-OS to track the line protocol or IPv4 or IPv6 routing state of an interface.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **track** *object-id* **interface** *interface-type number* { **ip routing** | **ipv6 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 512.

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
```

Copies the running configuration to the startup configuration.

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:

```
sswitch# configure terminal
switch(config)# track 2 interface ethernet 1/2 ip routing
switch(config-track)# copy running-config startup-config
```

This example shows how to configure object tracking for the IPv6 routing state on Ethernet 1/2:

```
switch# configure terminal
switch(config)# track 3 interface ethernet 1/2 ipv6 routing
switch(config-track)# copy running-config startup-config
```

Delete a tracking object

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **no track** *object-id*

Example:

```
switch(config)# no track 1
switch(config-track)#
```

Deletes a tracked object for an interface. The *object-id* range is from 1 to 512.

Step 3 (Optional) **copy running-config startup-config**

Example:

```
switch(config-track)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

This example shows how to delete a tracked object:

```
switch# configure terminal
switch(config)# no track 1
switch(config-track)# copy running-config startup-config
```

Configure object tracking for route reachability

You can configure Cisco NX-OS to track the existence and reachability of an IP route or an IPv6 route.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **track** *object-id* { **ip** | **ipv6** } **route** *prefix* / *length* **reachability**

Example:

```
switch(config)# track 3 ipv6 route 2::5/64 reachability
switch(config-track)#
```

Creates a tracked object for a route and enters tracking configuration mode. The *object-id* range is from 1 to 512. The prefix format for IPv4 is A.B.C.D/length, where the length range is from 1 to 32. The prefix format for IPv6 is A:B::C:D/length, where the length range is from 1 to 128.

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
```

Copies the running configuration to the startup configuration.

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
```

This example shows how to configure object tracking for an IPv6 route in the default VRF:

```
switch# configure terminal
switch(config)# track 5 ipv6 route 10::10/128 reachability
switch(config-track)# copy running-config startup-config
```

Configure 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

Step 1 **configure terminal**

Example:

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

Enters global 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 512.

Step 3 **object** *object-number* [**not**]

Example:

```
switch(config-track)# object 10
```

Adds a tracked object to the track list. The *object-id* range is from 1 to 512. 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** [*object-id*]

Example:

```
switch(config-track)# show track
```

Displays object tracking information.

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

Example:

```
switch(config-track)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

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
```

Configure 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

Step 1 **configure terminal**

Example:

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

Enters global 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 512.

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 is from 0 to 100 percent.

Step 4 **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 512.

Step 5 (Optional) **show track** [*object-id*]

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
```

Copies the running configuration to the startup configuration.

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
```

Configure 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

Step 1 **configure terminal**

Example:

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

Enters global 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 512.

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 is from 1 to 255.

Step 4 **object** *object-id* **weight** *value*

Example:

```
switch(config-track)# object 10 weight 15
```

Adds a tracked object to the track list. The *object-id* range is from 1 to 512. The *value* range is from 1 to 255. The default weight value is 10.

Step 5 (Optional) **show track** [*object-id*]

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
```

Copies the running configuration to the startup configuration.

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.

Configure 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 follows:

- 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

Step 1 **configure terminal**

Example:

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

Enters global 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 512. The prefix format for IPv4 is A.B.C.D/length, where the length range is from 1 to 32. The prefix format for IPv6 is A:B::C:D/length, where the length range is from 1 to 128.

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 512.

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.

The *track-number* range is from 1 to 512.

Step 5 (Optional) **show track** [*object-id*]

Example:

```
switch(config-track)# show track 3
```

Displays object tracking information.

Step 6 (Optional) **copy running-config startup-config**

Example:

```
switch(config-track)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

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
```

Configure object tracking for a nondefault VRF

Configure object tracking for a route in a nondefault VRF.

You can configure Cisco NX-OS to track an object in a specific VRF.

Before you begin

Ensure that nondefault VRFs are created first.

Procedure

Step 1 **configure terminal**

Example:

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

Enters global configuration mode.

Step 2 **track** *object-id* { **ip** | **ipv6** } **route** *prefix/length* **reachability**

Example:

```
switch(config)# track 3 ipv6 route 1::2/64 reachability
switch(config-track)#
```

Creates a tracked object for a route and enters tracking configuration mode. The *object-id* range is from 1 to 512. The prefix format for IPv4 is A.B.C.D/length, where the length range is from 1 to 32. The prefix format for IPv6 is A:B::C:D/length, where the length range is from 1 to 128.

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:

```
switch(config-track)# copy running-config startup-config
```

Copies the running configuration to the startup configuration.

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 configure object tracking for an IPv6 route and use VRF Red to look up reachability information for this object:

```
switch# configure terminal
switch(config)# track 3 ipv6 route 1::2/64 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
```

Verify the object tracking configuration

To display 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** | **ipv6** } **route** [**brief**]	Displays the IPv4 or IPv6 route-based object tracking information.

Configuration examples for object tracking

Object tracking configuration allows you to monitor the reachability of a route and associate the tracking with a specific VRF instance.

- Enables tracking of route reachability.
- Associates the tracked object with a VRF (Virtual Routing and Forwarding) instance.
- Configuration is saved to persist across reloads.

Object tracking configuration for route reachability using VRF

This section provides a configuration example for tracking the reachability of a route and associating it with a VRF instance.

- Configure object tracking for a specific route.
- Assign the tracked object to a VRF.
- Save the configuration.

1. Enter global configuration mode.
2. Configure object tracking for the route 209.165.201.0/8.
3. Associate the tracked object with VRF Red.
4. Save the running configuration to startup configuration.

Example: Configuring object tracking for route reachability with VRF Red

The following example shows the CLI commands to configure object tracking for route 209.165.201.0/8 and associate it with VRF Red:

```
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
```

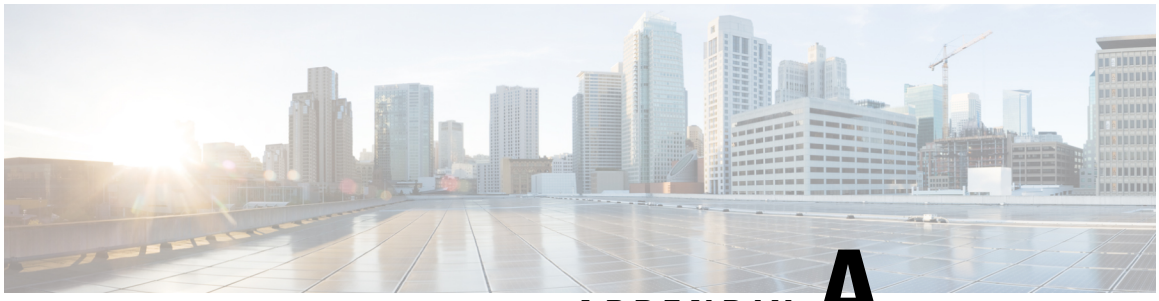
Related topics

- This topic provides references to related configuration topics for object tracking.
- Configuring Layer 3 Virtualization
- Configuring HSRP

Related documents

This section provides references to related configuration guides for Embedded Event Manager and IP SLA Object Tracking.

Related Topic	Document Title
Configuring the Embedded Event Manager	Cisco Nexus 9000 Series NX-OS System Management Configuration Guide
Configuring IP SLA Object Tracking	Cisco Nexus 9000 Series NX-OS IP SLAs Configuration Guide



APPENDIX A

IETF RFCs supported by Cisco NX-OS unicast features

This appendix lists the IETF RFCs for unicast routing supported in Cisco NX-OS.

- BGP RFCs, on page 693
- First-hop redundancy protocols RFCs, on page 694
- IP services RFCs, on page 695
- IPv6 RFCs, on page 695
- IS-IS RFCs, on page 696
- OSPF RFCs, on page 697
- RIP RFCs, on page 697

BGP RFCs

This section lists the RFCs and drafts relevant to BGP and their titles.

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 2545	*Use of BGP-4 Multiprotocol Extensions for iPv6 Inter-Domain Routing*
RFC 2858	*Multiprotocol Extensions for BGP-4*
RFC 2918	*Route Refresh Capability 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)*

RFCs	Title
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 4724	*Graceful Restart Mechanism for BGP*
RFC 4760	*Multiprotocol Extensions for BGP-4*
RFC 4781	*Graceful Restart Mechanism for BGP with MPLS*
RFC 4893	*BGP Support for Four-octet AS Number Space*
RFC 5004	*Avoid BGP Best Path Transitions from One External to Another*
RFC 5396 [1]	*Textual Representation of Autonomous System (AS) Numbers*
RFC 5549	*Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop*
RFC 5668	*4-Octet AS Specific BGP Extended Community*
RFC 7606	*Revised Error Handling for BGP Update Messages*
RFC 7854	*BGP Monitoring Protocol (BMP)*
draft-ietf-idr-add-paths-08.txt	*Advertisement of Multiple Paths in BGP*
draft-ietf-idr-bgp4-mib-15.txt	*BGP4-MIB*
draft-kato-bgp-ipv6-link-local-00.txt	*BGP4+ Peering Using IPv6 Link-local Address*
draft-ietf-idr-avoid-transition-05.txt	*Bestpath Transition Avoidance*
draft-ietf-idr-bgp4-mib-15.txt	*Peer Table Objects*
draft-ietf-idr-dynamic-cap-03.txt	*Dynamic Capability*

[1] RFC 5396 is partially supported. The asplain and asdot notations are supported, but the asdot+ notation is not.

First-hop redundancy protocols RFCs

This topic lists the RFCs and titles for first-hop redundancy protocols.

Table 49: First-Hop redundancy protocols RFCs

RFCs	Title
RFC 2281	*Hot Standby Redundancy Protocol*

RFCs	Title
RFC 3768	*Virtual Router Redundancy Protocol*
RFC 5798	*Virtual Router Redundancy Protocol (VRRP) Version 3 for IPv4 and IPv6*

IP services RFCs

This reference lists the RFCs relevant to IP services.

RFCs	Title
RFC 768	*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*
RFC 4022	*TCP-MIB*
RFC 4292	*IP-FORWARDING-TABLE-MIB*
RFC 4293	*IP-MIB*

IPv6 RFCs

This reference lists the key RFCs related to IPv6 and their titles.

RFCs	Title
RFC 1981	Path MTU Discovery for IP version 6
RFC 2374	An Aggregatable Global Unicast Address Format
RFC 2460	Internet Protocol, Version 6 (IPv6) Specification
RFC 2464	Transmission of IPv6 Packets over Ethernet Networks
RFC 3021	Using 31-Bit Prefixes on IPv4 Point-to-Point Links

RFCs	Title
RFC 4191	Default Router preferences and more specific routes
RFC 4193	Unique Local IPv6 Unicast Addresses **Note** RFC 4193 is partially supported. Section 3.2.2 is not supported.
RFC 4291 (replaced RFC 2373)	IP Version 6 Addressing Architecture
RFC 4443 (replaced RFC 2463)	ICMPv6
RFC 4861 (replaced RFC 2461)	Neighbor Discovery for IP Version 6 (IPv6)
RFC 4862 (replaced RFC 2462)	IPv6 Stateless Address Autoconfiguration
RFC 6106	IPv6 Router Advertisement Options for DNS Configuration
RFC 2526	Reserved IPv6 Subnet Anycast Addresses

IS-IS RFCs

This topic lists the RFCs and Internet Drafts relevant to IS-IS.

Table 50: IS-IS RFCs and Internet Drafts

RFCs	Title
RFC 1142	*OSI 10589 intermediate system to intermediate system intro-domain routing exchange protocol*
RFC 1195	*Use of OSI IS-IS for routing in TCP/IP and dual environment*
RFC 2763, RFC 5301	*Dynamic Hostname Exchange Mechanism for IS-IS*
RFC 2966, RFC 5302	*Domain-wide Prefix Distribution with Two-Level IS-IS*
RFC 2972	*IS-IS Mesh Groups*
RFC 3277	*IS-IS Transient Blackhole Avoidance*
RFC 3373, RFC 5303	*Three-Way Handshake for IS-IS Point-to-Point Adjacencies*
RFC 3567, RFC 5304	*IS-IS Cryptographic Authentication*
RFC 3784, RFC 5305	*IS-IS Extensions for Traffic Engineering*
RFC 3847, RFC 5306	*Restart Signaling for IS-IS*
RFC 4205, RFC 5307	*IS-IS Extensions in Support of Generalized Multi-Protocol Label Switching*

RFCs	Title
draft-ietf-isis-igp-p2p-over-lan-06.txt	*Internet Draft Point-to-point operation over LAN in link-state routing protocols*

OSPF RFCs

This section lists the key RFCs and drafts related to OSPF and its extensions.

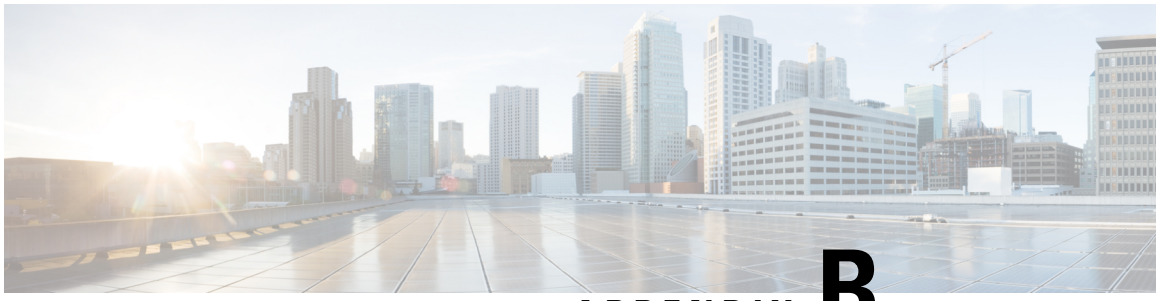
Table 51: OSPF RFCs and Drafts

RFCs	Title
RFC 2328	*OSPF Version 2*
RFC 2370	*The OSPF Opaque LSA Option*
RFC 2740	*OSPF for IPv6*
RFC 3101	*The OSPF Not-So-Stubby Area (NSSA) Option*
RFC 3137	*OSPF Stub Router Advertisement*
RFC 3623	*Graceful OSPF Restart*
RFC 5709	*OSPFv2 HMAC-SHA Cryptographic Authentication*
draft-ietf-ospf-ospfv3-graceful-restart-04.txt	*OSPFv3 Graceful Restart*

RIP RFCs

This reference lists the RFCs relevant to RIP.

RFCs	Title
RFC 2082	*RIP-2 MD5 Authentication*
RFC 2453	*RIP Version 2*



APPENDIX B

Configuration Limits for Cisco NX-OS Layer 3 Unicast Features

- Configuration limits for Cisco NX-OS layer 3 unicast features, on page 699
- Layer 3 heavy routing template, on page 699

Configuration limits for Cisco NX-OS layer 3 unicast features

The configuration limits for Cisco NX-OS layer 3 unicast features specify the maximum supported values for various configuration parameters. Limits are documented in the Cisco Nexus 9000 Series NX-OS Verified Scalability Guide.

For detailed configuration limits, refer to the official scalability guide.

- Cisco Nexus 9000 Series NX-OS Verified Scalability Guide

Layer 3 heavy routing template

The Layer 3 Heavy Routing Template is a system routing profile that reallocates hardware resources to increase the maximum number of IPv4 and IPv6 unicast routes supported. It dedicates more hardware to efficient storage and lookup of Layer 3 prefixes using a routing trie structure.

- Optimizes hardware for high-scale Layer 3 routing environments.
- Enables greater route scale by increasing trie-based route storage.

Use cases and benefits

This template is intended for situations where a large volume of IPv4 and IPv6 routes must be supported, such as:

- Data center or cloud deployments with extensive routing requirements.
- Environments where the default routing profile does not provide adequate route capacity.

Guidelines and limitations for layer 3 heavy routing template

Use the Layer 3 Heavy Routing Template only on supported cloud-scale hardware. The template is not compatible with platforms outside the supported list and may impact available hardware resources for other features.

- You can configure the Layer 3 Heavy Routing Template on FX, FX2, FX3, GX, GX2, and HX cloud-scale Cisco Nexus 9000 switches.
- Layer 3 Heavy Routing Template is not supported on other cloud-scale TOR or EOR platforms.
- Template changes take effect only after saving the configuration and reloading the switch.
- Enabling this template reallocates hardware tiles to maximize route scale, which may reduce resources available for less-used features.
- Increased hardware allocation for Layer 3 routing may result in higher memory consumption.

Layer 3 heavy routing template scale limits

This section lists scale limits and technical specifications for the Layer 3 Heavy Routing Template compared to the default routing template, as provided in the engineering documentation.

The following table lists:

- **Resource (entries):** The hardware resource or route type being measured.
- **Default Template:** Maximum value supported with the default routing template.
- **Layer 3 Heavy Template:** Maximum value supported with the Layer 3 Heavy Routing Template.

Table 52: Layer 3 heavy routing template vs. default template: scale limits

Resource	Default Template	Layer 3 Heavy Template
Max IPv4 Trie route entries	1,048,576	1,256,448
Max IPv6 Trie route entries	589,824	628,224
Max TCAM table entries	24,576	24,576
Max V4 Ucast TCAM table entries	16,384	16,384
Max V6 Ucast TCAM table entries	2,048	2,048
Max native host route entries (shared v4/v6)	196,608	65,536
Max Nexthop table entries	98,304	98,304
Max ARP entries	98,304	65,536
Max ND entries	98,304	32,768

Configure the layer 3 heavy routing template

This task allows you to change the system routing profile to maximize Layer 3 route scale using the Layer 3 Heavy Routing Template.

- Support larger IPv4 and IPv6 routing tables in hardware.
- Resolve route scale limitations due to default hardware resource allocation.

Use this procedure if your network requires greater Layer 3 route capacity than the default system routing template provides.

Before you begin

Ensure the following before performing this task:

- Save your current configuration and schedule a maintenance window to reload the switch.
- Review the memory and resource impact before enabling this feature.

Procedure

Step 1 Use the **system routing template-l3-heavy** command to apply the Layer 3 Heavy Routing Template.

Example:

```
switch(config)# system routing template-l3-heavy
```

This command applies the Layer 3 Heavy Routing Template and increases the hardware allocation for route storage.



Note You must save the configuration and reload the switch for the change to take effect.

The system is configured to use the Layer 3 Heavy Routing Template after the next reload.

Step 2 Use the **copy running-config startup-config** command to save the configuration.

Example:

```
switch(config)# copy running-config startup-config
```

This ensures your new template configuration persists across a reload.

Configuration is saved to startup-config.

Step 3 Use the **reload** command to reload the switch.

Example:

```
switch# reload
```

The switch must be reloaded for the new routing template to take effect.

Layer 3 Heavy Routing Template becomes active after reload.

What to do next

After completing the configuration, verify that the Layer 3 Heavy Routing Template is active and functioning as expected.

See Verify Layer 3 Heavy Routing Template for verification steps.

Verify layer 3 heavy routing template

This verification ensures the Layer 3 Heavy Routing Template is correctly applied and that the system is operating in the intended routing mode. Validate that hardware resources are allocated to maximize Layer 3 route scale.

After enabling the Layer 3 Heavy Routing Template and reloading the switch, use these steps to verify that the configuration is active and functioning as expected.

Before you begin

Before starting verification, make sure the Layer 3 Heavy Routing Template is configured and the switch has been reloaded to apply the changes.

Procedure

Step 1 Use the **show system routing mode** command to check the system routing mode configuration and applied state.

Example:

```
switch#
show system routing mode
Configured System Routing Mode: L3 Heavy
Applied System Routing Mode: L3 Heavy
```

The command output should display **Configured System Routing Mode: L3 Heavy** and **Applied System Routing Mode: L3 Heavy** .

The system confirms that the Layer 3 Heavy Routing Template is both configured and applied.

Step 2 Use the **show hardware internal forwarding table utilization module 1** command to verify hardware forwarding table resource allocation.

Example:

```
switch#
show hardware internal forwarding table utilization module 1
IPv4/IPv6 hosts and routes summary on module : 1
------------------------------------------------
Configured System Routing Mode: NX_L3_Heavy
Max IPv4 Trie route entries: 1153433
Max IPv6 Trie route entries: 628224
```

Check that the **Configured System Routing Mode" is "NX_L3_Heavy** and review the maximum values for IPv4 and IPv6 trie route entries, TCAM table entries, and other relevant hardware resource limits.

Values should match the expected scale for the Layer 3 Heavy Routing Template.

Hardware resources have been reallocated to support increased Layer 3 route scale.

Step 3 (Optional) Use the **show hardware internal tah logical-tile-types** command to verify logical tile allocation details.

Example:

```
            switch#
            show hardware internal tah logical-tile-types
            Logical Tile Types
            Instance : 0
            Profile: L3 Heavy Scale
            FPX: 0 Tile: 0 Tile-depth: 8192 type_tile_id: 0/0 lo: 0x0 col: 1 offset: 0x0 TT: IP
Log.TT: IP DS:Hash table TP:Left tilepair v4/v6: 0/0 htype: 0
```

This command provides advanced detail on how hardware tiles are allocated to different functions under the active routing template.

Logical tile allocation reflects the Layer 3 Heavy Routing Template settings.

The switch is confirmed to be operating with the Layer 3 Heavy Routing Template and supporting the intended route scale and hardware allocation.

Configuration Limits for Cisco NX-OS Layer 3 Unicast Features



INDEX

{ip | ipv6} 540
{ip | ipv6} address 280
{ip | ipv6} bandwidth eigrp 244
{ip | ipv6} bandwidth-percent eigrp 244
{ip | ipv6} delay eigrp 244
{ip | ipv6} distribute-list eigrp 244
{ip | ipv6} hello-interval eigrp 241
{ip | ipv6} hold-time eigrp 241
{ip | ipv6} next-hop-self eigrp 245
{ip | ipv6} passive-interface eigrp 245
{ip | ipv6} prefix-list 540
{ip | ipv6} router eigrp 246
{ip | ipv6} router isis 263, 280
{ip | ipv6} split-horizon eigrp 242

A

additional-paths receive 371
additional-paths selection route-map 373
additional-paths send 371
address 656, 666, 668, 671
address-family 231, 235, 382
address-family {ipv4 | ipv6} {unicast | multicast 414
address-family {ipv4 | ipv6} {unicast | multicast} 380, 389, 391
address-family {ipv4 | ipv6} unicast 238, 240, 270–271, 405
address-family {ipv4|ipv6} {multicast|unicast} 342, 344, 348
address-family {ipv4|ipv6} {unicast|multicast} 309, 312
address-family ipv4 unicast 276, 451, 456
address-family ipv6 unicast 178–179, 181, 186, 188–189, 191, 194, 277, 451, 456, 469, 475
adjacency-check 282
advertise-active-only 342
advertise-map 389
advertisement interval 660
aggregate-address 302
allowas in 411
area 127, 131–135, 137, 143, 178–181, 184, 189
as-override 412
authentication 138
authentication key-chain 231, 265
authentication mode md5 231
authentication text 659
authentication-check {level-1 | level-2} 265
authentication-key 138
authentication-type {cleartext | md5} {level-1 | level-2} 265
autonomous-system 227, 238

B

bgp confederation peers 379

C

capability additional-paths receive 370
capability additional-paths send 370
clear bgp 318
clear bgp {ipv4 | ipv6} {unicast | multicast 365–366
clear bgp {ipv4 | ipv6} {unicast | multicast} flap-statistics [vrf 318
clear bgp all 318
clear bgp all dampening 318
clear bgp all flap-statistics 318
clear bgp dampening 320
clear bgp flap-statistics 320
clear forwarding 517
clear ip eigrp redistribution 237
clear ip mbgp 320
clear ip mbgp dampening 321
clear ip mbgp flap-statistics 321
clear ip rip statistics 463
clear ipv6 rip statistics 478
clear isis 261
clear rip policy statistics redistribute 463
clear vrrpv3 statistics 673
client-to-client reflection 380
cluster-id 380
confederation identifier 379
configure terminal 565
configuring 626, 628
  HSRP master group 626
  HSRP slave group 628
continue 551

D

dampening 384
dead-interval 138, 184
default settings 468
  RIPng 468
default-information originate 140, 186, 243, 271–272, 405, 456

default-metric 141, 186, 235, 392, 456
default-originate 411
delay 688
delay restore 170
description 312, 340, 410, 666
disable-connected-check 313, 374
disable-peer-as-check 411
disable-policy-batching 370
distance 124, 174, 243, 260, 410, 452, 470
distribute {level-1 | level-2} into {level-1 | level-2} 271–272
distribute-list 248
dont-capability-negotiate 369
dscp 386
dynamic routing protocols 14

E

ebgp-multihop 313, 377
enforce-first-as 409
enhanced-error 407

F

feature bgp 308
feature eigrp 225
feature hsrp 622
feature interface vlan 483
feature isis 259
feature ospf 121
feature ospfv3 172
feature pbr 563–565, 574
feature rip 451, 468
feature vrrp 655
feature vrrpv3 665
filter-list 412
flush-routes 229

G

gateway protocols 14
graceful restart 278
graceful-restart 150, 196, 240, 438
graceful-restart grace-period 150, 196
graceful-restart helper-disable 151, 196
graceful-restart planned-only 151, 196
graceful-restart t3 manual 278
graceful-restart-helper 438

H

hardware ip glean throttle 51–52
hardware ip glean throttle maximum 52
hardware ip glean throttle maximum timeout 53
hello-interval 138, 184
hostname dynamic 268
hsrp 623, 625, 631, 633
HSRP 614
    multiple group optimization 614
HSRP master group 626
    configuring 626
HSRP slave group 628
    configuring 628
hsrp timers extended-hold 638
hsrp version {1 | 2} 622
hsrp version 2 625

I

inherit peer 345, 348
inherit peer-policy 342, 344
inherit peer-session 340, 344
interface 231
interface ethernet 35–36
interface-vlan 483
ip 223, 623, 625
ip | ipv6} offset-list eigrp 245
ip address 35–36, 125, 153, 483, 495, 497, 668, 671
ip arp address 44
ip arp gratuitous {request | update} 48
ip as-path access-list 543
ip authentication key-chain eigrp 232
ip authentication mode eigrp 232
ip autoconfig 625
ip community-list expanded 548
ip community-list standard 548
ip directed-broadcast 51
ip domain-list 106, 108, 499
ip domain-lookup 106
ip domain-name 106–107
ip extcommunity-list expanded 549
ip extcommunity-list standard 549
ip host 105
ip name-server 106, 108
ip ospf authentication 128
ip ospf authentication key-chain 128
ip ospf authentication-key 127, 129
ip ospf cost 125
ip ospf dead-interval 125, 149
ip ospf hello-interval 126, 149
ip ospf message-digest-key 127, 129
ip ospf mtu-ignore 126
ip ospf passive-interface 126
ip ospf retransmit-interval 149
ip ospf transmit-delay 149
ip passive-interface eigrp 230
ip proxy arp 45
ip rip authentication keychain 454
ip rip authentication mode 454
ip rip passive-interface 455
ip rip poison-reverse 455

ip rip summary-address 455
ip route 404, 482–483, 485, 494
ip router eigrp 227, 232
ip router ospf 125, 153, 497
ip router rip 453
ip source 53
ip summary-address eigrp 234
ip tcp path-mtu-discovery 50
IPv4 55
  related documents 55
ipv6 223
ipv6 address 82, 175, 198, 475, 625
ipv6 address use-link-local-only 82
ipv6 authentication key-chain eigrp 232
ipv6 authentication mode eigrp 232
ipv6 ospfv3 198
ipv6 passive-interface eigrp 230
ipv6 rip poison-reverse 472
ipv6 rip route-filter 477
ipv6 rip summary-address 455
ipv6 route 482, 485
ipv6 router eigrp 227, 232
ipv6 router ospfv3 175, 183
ipv6 router rip 471, 475
ipv6 summary-address eigrp 234
is-type {level-1 | level-2 | level-1-2} 260
isis authentication key-chain 266
isis authentication-check {level-1 | level-2} 266
isis authentication-type {cleartext | md5} {level-1 | level-2} 266
isis circuit-type {level-1 | level-2 | level-1-2} 263
isis csnp-interval 282
isis hello-interval 282
isis hello-multiplier 282
isis hello-padding 269
isis lsp-interval 283
isis mesh-group 267
isis metric 263
isis passive {level-1 | level-2 | level-1-2} 263
isis priority 267
isis shutdown 264

K

key 127–128

L

link-state protocols 15
local-as 378
log-adjacency-changes 123–124, 174, 227, 260
log-neighbor-changes 409–410
log-neighbor-warnings 227
low-memory exempt 410
lsp-gen-interval 281
lsp-mtu 261

M

mac address 671
mac-address 630
match ip address prefix-list 146
match ip route-source 170
match ip route-source prefix-list 146, 170, 192
match ipv6 address 170
match ipv6 address prefix-list 170, 192
match ipv6 route-source 170
match route-type 146, 170, 191
match-address 666
max-lsp-lifetime 282
max-metric router-lsa 144
maxas-limit 377
maximum routes 518
maximum-paths 123–124, 152, 174, 198, 239, 261, 384, 452, 470, 497
maximum-peers 346
maximum-prefix 342
medium {broadcast | p2p} 263
message-digest-key 138
metric direct 0 458, 473
metric max-hops 243
metric weights 243
metrics rib-scale 242
metrics version 64bit 242
MGO 614
  HSRP 614

N

name 635
neighbor 314, 340, 342, 345, 348, 380, 382, 389, 414, 440
neighbor-down fib-accelerate 369
net 260, 280
network 309
next-hop-self 366, 381
next-hop-third-party 366
nexthop suppress-default-resolution 367–368
no {ip | ipv6} route 482
no adjacency-check 277
no adjacency-checkg 276
no fast-external-fallover 377
no preempt 661
no shutdown 623, 626, 656–657, 659–661, 663
no track 682
nsf await-redist-proto-convergence 244

O

object 684–686
ospfv3 cost 176
ospfv3 dead-interval 176
ospfv3 hello-interval 176
ospfv3 instance 176

ospfv3 mtu-ignore 176
ospfv3 network 176
ospfv3 passive-interface 176
ospfv3 priority 176
ospfv3 retransmit-interval 194
ospfv3 transmit-delay 195

P

passive-interface default 124, 174
password 339
path-attribute discard 406
path-attribute treat-as-withdraw 406
preempt 634–635, 666
prefix-list 412
priority 633, 635, 657, 667

R

redistribute 140, 186, 235, 237, 271–273, 456
redistribute {direct | {eigrp | isis | ospf | ospfv3 | rip} 391
redistribute bgp 188
redistribute maximum-prefix 188, 237, 273
redistribute static route-map allow 405
reference-bandwidth 261
related documents 55
  IPv4 55
reload 37–38, 40, 84–87, 317
reload module 117, 168, 256
remove-private-as 302, 411
restart bgp 310
restart eigrp 229
restart isis 256, 262
  IS-IS 262
restart ospf 117, 151
restart ospfv3 168, 197
restart rip 470
retransmit-interval 139, 184
RIPng 468, 476–477
  default settings 468
  enabling 468
  tuning 476
  verifying 477
RIPng statistics 478
  displaying 478
route-map 146, 191, 383, 545–546, 551
route-map allow permit 404
route-reflector-client 380, 382
router bgp 309, 311, 313, 339, 341, 344, 347, 379, 382, 389, 391, 404, 414, 437, 440
router eigrp 226, 231, 235, 237–238, 240, 246
router isis 260, 265, 270–271, 273, 276, 278–279
router ospf 127, 131–132, 134, 137, 140, 143–145, 148, 150, 152, 496
router ospfv3 173, 178–179, 181, 184, 186, 188–189, 191, 193, 196, 198
router rip 451, 456, 458, 469, 473–474
router-id 174, 309, 409
routing-context vrf 499
routing-context vrf default 500

S

send-community 412
send-community extended 412
set distance 146, 192
set ip next-hop peer-address 381
set ipv6 next-hop peer-address 381–382
set next-hop 381
set-attached-bit 269
set-overload-bit {always | on-startup 268
show 499, 515
show {ip | ipv6} 228, 485, 540, 543
show {ip | ipv6} adjacency 520
show {ip | ipv6} eigrp 247–248
show {ip | ipv6} eigrp route-map statistics redistribute 235
show {ip | ipv6} route 515, 520
show {ip | ipv6} routing 515
show {ip | ipv6} static-route 482, 486
show {ip | ipv6} static-route track-table 486
show {ipv | ipv6} bgp 323
show {ipv | ipv6} mbgp 323
show {ipv4 | ipv6} bgp 443
show {ipv4 | ipv6} mbgp 443
show bgp 441
show bgp {ipv4 | ipv6 | vpnv4 | vpnv6} {unicast | multicast} 441–442
show bgp {ipv4 | ipv6} {unicast | multicast} 321–323, 383, 441–443
show bgp {ipv4 | ipv6} {unicast | multicast} neighbors 381
show bgp {ipv4 | ipv6} unicast 373, 407
show bgp {ipv4 | ipv6} unicast injected-routes 443
show bgp {ipv4 | ipv6} unicast path-attribute discard 407
show bgp {ipv4 | ipv6} unicast path-attribute unknown 407
show bgp {ipv4|ipv6} {unicast|multicast} neighbors 312, 314
show bgp {ipv4|ipv6} unicast neighbors 348, 442
show bgp all 309, 321, 441
show bgp convergence 321, 441
show bgp ipv4 multicast neighbors 390
show bgp ipv4 unicast neighbors 347, 390
show bgp ipv6 multicast neighbors 390
show bgp ipv6 unicast neighbors 390
show bgp neighbor 341, 343, 345, 370–371
show bgp paths 322
show bgp peer-policy 323, 343, 442
show bgp peer-session 323, 340, 442
show bgp peer-template 323, 345, 442
show bgp process 323, 442
show bgp sessions 323, 443
show bgp statistics 323, 443
show bgp vrf 321
show consistency-checker 516
show feature 121, 172, 226, 259, 308, 451, 468, 563–564
show fhrp 667, 669, 672

show forwarding 516
show forwarding {ip | ipv4 | ipv6} route 520
show forwarding adjacency 520
show forwarding distribution {clients | fib-state} 520
show forwarding interfaces module 520
show forwarding route summary 37–40, 84–87
show hosts 106, 108–109
show hsrp 624, 626, 632, 638
show hsrp delay interface 638
show hsrp group 638–639
show hsrp interface 634, 639
show interface 672
show ip adjacency 54
show ip adjacency summary 55
show ip arp 55
show ip arp statistics 55
show ip arp summary 55
show ip bgp neighbors 348, 442
show ip community list 548
show ip community-list 550, 553
show ip eigrp neighbor detail 233
show ip ext community-list 553
show ip interface 36, 55
show ip load-sharing 513
show ip ospf 125, 127, 129, 133, 135, 149, 151
show ip ospf interface 126
show ip ospf neighbor 126
show ip ospf policy statistics area 131, 155
show ip ospf statistics 155
show ip ospf summary-address 143
show ip ospf traffic 155
show ip ospf virtual-link 138
show ip policy statistics redestribute 155
show ip rip 452–453, 463, 469
show ip rip route 457
show ip route 484
show ipv6 adjacency 100–101
show ipv6 interface 82, 100–101
show ipv6 ospfv3 174–175, 183, 196
show ipv6 ospfv3 memory 211
show ipv6 ospfv3 policy statistics area 178, 211
show ipv6 ospfv3 policy statistics redistribute 211
show ipv6 ospfv3 statistics 211
show ipv6 ospfv3 summary-address 190
show ipv6 ospfv3 traffic 211
show ipv6 ospfv3 virtual-link 184
show ipv6 rip 471, 475, 477–478
show ipv6 rip instance 477
show ipv6 routers interface 348, 443
show ipv6 static-route vrf 486
show isis 260, 263, 270–272, 281, 283–284
show platform fib 18
show platform forwarding 18
show policy 576
show prefix-list 553
show route-map 553, 576
show route-map brief 553
show routing 519–520
show routing hash 513
show running-config bgp 439
show running-config eigrp 237
show running-config isis 273–274, 277–278
show running-config ospfv3 188
show running-config rip 459, 473
show running-configuration bgp 323, 443
show running-configuration eigrp 247
show running-configuration isis 284
show running-configuration rip 478
show tech-support isis 284
show track 681, 683–685, 687–688, 690–691
show vrf 494, 496, 500
show vrrp 656–657, 659–660, 662–663, 672
show vrrp statistics 673
show vrrpv3 667, 669
show vrrpv3 statistics 673
show vrrs pathway 671
shutdown 229, 262, 310, 312, 657–658, 660–661, 663, 667, 669
snmp-server host 498
soft-reconfiguration inbound 365
spf-interval [level-1 | level-2 282
static routes 14
stub 233
stub routing 13
summary-address 143, 189, 270
suppress-fib-pending 302, 388
suppress-inactive 412
system pic enable 317
system pic-core 317
system routing max-mode host 37, 84
system routing max-mode l3 40, 87
system routing mode hierarchical 64b-alpm 39, 86
system routing non-hierarchical-routing 38, 85
system switchover 117, 168, 256

T

table-map 145, 191
template peer 344
template peer-session 339, 342
test forwarding 516
threshold percentage up 685
threshold weight up 686
timers 312, 340, 344–345, 635
timers [bestpath-delay 409
timers active-time 244
timers advertise 667
timers basic 477
timers lsa-arrival 148, 194
timers lsa-group-pacing 148, 194
timers nsf converge 240
timers nsf route-hold 240

timers nsf signal 240
timers prefix-peer-timeout 346, 438
timers throttle lsa 148, 194
timers throttle spf 148
track 633–634, 681, 683–686, 688–689
track interface 663
transmit-delay 139, 184
transport connection-mode passive 410
tuning 476
RIPng 476

U

update-source 382, 411

V

verifying 477
RIPng 477
vrf 152, 198, 280, 440, 475, 497
vrf context 107, 152, 198, 246, 279, 440, 474, 485, 494, 499, 518
IS-IS 279
vrf member 153, 198, 246, 280, 475, 495, 497, 690
vrrp 656–658, 660–661, 663
vrrp2 667
vrrpv3 666, 668
vrrs leader 667
vrrs pathway 671

W

weight 312
write erase 492
write erase boot 492