CONFIGURATION GUIDE

Extreme SLX-OS Layer 3 Routing Configuration Guide, 18s.1.03a

Supporting the ExtremeSwitching SLX 9140 and SLX 9240 Switches

9036273-01 Rev AA
June 2020

Copyright © 2020 Extreme Networks, Inc. All Rights Reserved.

Legal Notice

Extreme Networks, Inc. reserves the right to make changes in specifications and other information contained in this document and its website without prior notice. The reader should in all cases consult representatives of Extreme Networks to determine whether any such changes have been made.

The hardware, firmware, software or any specifications described or referred to in this document are subject to change without notice.

Trademarks

Extreme Networks and the Extreme Networks logo are trademarks or registered trademarks of Extreme Networks, Inc. in the United States and/or other countries.

All other names (including any product names) mentioned in this document are the property of their respective owners and may be trademarks or registered trademarks of their respective companies/owners.

For additional information on Extreme Networks trademarks, see: www.extremenetworks.com/company/legal/trademarks

Open Source Declarations

Some software files have been licensed under certain open source or third-party licenses. End-user license agreements and open source declarations can be found at: https://www.extremenetworks.com/support/policies/open-source-declaration/

Contents

Preface 21
- Conventions 21
  - Notes, cautions, and warnings 21
  - Text formatting conventions 21
  - Command syntax conventions 22
- Documentation and Training 22
- Getting Help 22
  - Subscribe to Service Notifications 23
- Providing Feedback 23

About This Document 25
- Supported hardware and software 25
- What's new in this document 25

ARP 27
- ARP and Neighbor Discovery overview 27
- Basic ARP configuration 27
  - Modifying the ARP aging timeout 27
  - Enabling and disabling ARP learning 28
  - Creating static ARP entries 28
  - Creating a VRF static ARP entry 30
  - Proxy ARP 30
  - ARP show and clear commands 31
- Dynamic ARP Inspection (DAI) 32
  - ARP poisoning 32
  - DAI functionality 32
  - DAI implementation 33
  - DAI show and clear commands 37
- ARP Guard 38
  - ARP Guard and DAI 38
  - ARP Guard implementation 39
  - ARP Guard show commands 42
- ARP and ND suppression 42
  - Supported platforms 43
  - Enabling ARP and ND suppression on a VLAN 43
  - Enabling ARP and ND suppression on a bridge domain 43
  - ARP and ND suppression show and clear commands 44
- Conversational ARP and ND 45
  - Supported platforms 46
  - Enabling and disabling conversational ARP and ND 46
  - Conversational ARP and ND show and clear commands 46

IP Addressing 49
- IP addressing overview 49
  - IPv4 address classes 49
  - IPv4 address classes 50
  - IP Interfaces 50
  - IP Interfaces 50

The ARP cache....50
31-bit subnet masks on point-to-point networks....51
IP unnumbered interfaces....51
Basic IP parameters and defaults....51
Parameter changes in effect....52
IP global parameters....52
IP interface parameters....57
Assigning IP addresses to interfaces....59
Assigning an IP address to a loopback....59
Assigning IP addresses to Ethernet interfaces....60
Deleting an IP address from an interface....60
IP directed broadcast on an interface....61
Enabling IP directed broadcast on an interface....61
Router-ID IP addresses....61
Assigning a router-ID IP address....62
DNS....62
DNS Server....62
DNS Resolver....62
DNS Gateway Addresses....63
Configuring DNS....63
Source IP address for various packet types....64
IPv4 MTU settings....65
IPv4 MTU and maximum frame size....65
Changing IPv4 MTU on an interface....65
Configuring the maximum ICMP error message rate....66
Disabling ICMP messages....67
Disabling ICMP redirect messages....68
IP addressing Show and Clear commands....69

IPv6 Addressing....71
IPv6 addressing overview....71
Configuring an IPv6 address....72
IPv6 management....72
Configuring IPv6 management ACLs....73
Specifying an IPv6 SNMP trap receiver....73
Secure Shell, SCP, and IPv6....73
IPv6 Telnet....73
Establishing a Telnet session from an IPv6 host....74
IPv6 traceroute....74
Restricting Web management access....74
IPv6 neighbor discovery configuration....74
IPv6 neighbor discovery configuration notes....75
Neighbor solicitation and advertisement messages....75
Router advertisement and solicitation messages....75
Neighbor redirect messages....76
Duplicate Address Detection (DAD)....76
Configuring IPv6 static neighbor entries....77
IPv6 router advertisement parameters....77
Setting IPv6 router advertisement parameters....77
Flags in IPv6 router advertisement messages....78
Setting flags in IPv6 router advertisement messages....78

Sending or suppressing IPV6 router advertisements and addresses 79
IPv6 router advertisement preference support 79
Configuring IPv6 RA preference 79
Reachable time for remote IPv6 nodes 80
Configuring reachable time for remote IPv6 nodes 80
Configuring an IPv6 domain name and server 80
IPv6 MTU 80
Changing IPv6 MTU 81
Information about IPv6 prefix list 81
Configuring an IPv6 prefix list 82
Displaying prefix list information 82
Displaying global IPv6 information 82
Displaying IPv6 interface information 83
Displaying IPv6 neighbor information 83
Displaying the IPv6 route table 84
Clearing global IPv6 information 84
Clearing IPv6 neighbor information 84
Clearing IPv6 routes from the IPv6 route table 84
Prevention of DDoS attack on IPv6 subnet-router anycast address 85
Enabling hardware drop of IPv6 subnet-router anycast address traffic 85

IPv4 Static Routing 87
Overview of static routing 87
Static route states follow port states 88
Configuring a basic IP static route 89
Adding metrics to a static route 90
Naming an IP static route 92
Removing a name or a static route 92
Configuring a physical interface as next hop 92
Configuring a virtual interface as next hop 94
Configuring a static route with a VRF as next hop 95
Configuring an MPLS next hop for an IPv4 static route 96
Configuring a static route for use with a route map 97
Configuring a null route 98
Dropping traffic to the null route in hardware 99
Configuring CAM default route aggregation 100
Configuring a default static route 100
Resolving a static route using other static routes 101
Resolving the next hop through a protocol 102
Static routes between VRFs 103
Configuring a static route from the default VRF to a non-default VRF 103
Configuring a static route from a non-default VRF to the default VRF 104
Configuring a static route between two named VRFs 104
Configuring an IP static interface route across VRFs 105
Configuring load sharing and redundancy 108
Displaying IPv4 static routes 111

IPv6 Static Routing 115
Overview of static routing 115
Static route states follow port states 116
Configuring a basic IPv6 static route 117

Naming an IPv6 static route....118
Removing an IPv6 static route....118
Removing an IPv6 static route in a non-default VRF....121
Configuring an interface as next hop....122
Configuring a virtual interface as next hop....124
Configuring a tunnel as next hop....125
Configuring a VRF as next hop for an IPv6 static route....126
Adding metrics to an IPv6 static route....127
Configuring a null route....129
Configuring a default static route....131
Resolving the IPv6 static route through a protocol....132
Configuring load sharing and redundancy....132
Adding an IPv6 static route tag for use with route-maps....134
IPv6 multicast static routes....134
Configuring IPv6 multicast routes in a non-default VRF....135
Displaying information on IPv6 static routes....136

DHCPv4....139
DHCPv4 overview....139
IP DHCP Relay function....139
Configuring IP DHCP Relay....141
DHCP relay agent supported user scenarios....143
DHCP server local....143
DHCP server remote....143
DHCP servers in a network....143
Multiple DHCP servers....143
DHCP server in different VRFs....144
DHCP relay agent information option 82....144
Enabling the DHCP Relay Agent Information option....145
Configuring DHCP relay gateway address....146
Configuring DHCP relay gateway interface....147
VRF support....147
Supported VRF configuration examples....148
VRF configuration examples to avoid....148
Displaying IP DHCP Relay statistics....148
Displaying IP DHCP Relay addresses on specific devices....150
Displaying IP DHCP relay addresses for an interface....153
Clearing IP DHCP relay statistics....154

DHCPv6....157
DHCPv6 overview....157
DHCP relay agent for IPv6....157
DHCPv6 relay agent....157
DHCPv6 multicast addresses and UDP ports....158
Multicast addresses....158
UDP ports....159
DHCPv6 address assignment....159
Basic DHCPv6 relay assignment....159
DHCPv6 prefix delegation....159
Relay chaining....159
Relay-message option....159

Remote-ID option 160
Interface-ID option 160
DHCPv6 message format 160
DHCPv6 relay provisioning 161
Configuring IPv6 DHCP relay 162
Displaying DHCPv6 relay addresses on specific devices 163
Displaying DHCPv6 relay addresses for an interface 165
Displaying IPv6 DHCP relay statistics 166
Clearing IP DHCPv6 relay statistics 168

BFD 169
Bidirectional Forwarding Detection overview 169
General BFD considerations and limitations 170
BFD on Network OS hardware platforms 171
BFD for Layer 3 protocols 173
BFD considerations and limitations for Layer 3 protocols 175
BFD for multihop ECMP 176
BFD for Layer 3 protocols on virtual Ethernet interfaces 176
BFD support for Layer 3 protocols on VEs over VCS 176
BFD for Layer 3 protocols on vLAGs 177
BFD for L3 protocols on port channels 177
Configuring BFD on an interface 178
Configuring BFD on an interface 178
Disabling BFD on an interface 179
BFD for BGP 180
BFD for BGP session creation and deletion 181
Configuring BFD session parameters for BGP 182
Configuring BFD session parameters for BGP 182
Enabling BFD sessions for a specified BGP neighbor 183
Enabling BFD sessions for a specified BGP neighbor 184
Enabling BFD sessions for a specified BGP neighbor in a nondefault VRF 185
Enabling BFD sessions for a specified BGP neighbor in a nondefault VRF 186
Enabling BFD sessions for a specified BGP peer group 186
Enabling BFD sessions for a specified BGP peer group 187
Enabling BFD sessions for a specified BGP peer group in a nondefault VRF 188
Enabling BFD sessions for a specified BGP peer group in a nondefault VRF 189
BFD for OSPF 190
BFD for OSPF session creation and deletion 190
Enabling BFD on a specified OSPFv2-enabled interface 191
Configuring BFD for OSPFv2 globally 192
Configuring BFD for OSPFv2 globally in a nondefault VRF instance 193
Enabling BFD on a specified OSPFv3-enabled interface 193
Enabling BFD on a specified OSPFv3-enabled interface 194
Configuring BFD for OSPFv3 globally 194
Configuring BFD for OSPFv3 globally in a nondefault VRF instance 195
BFD for IS-IS 196
Enabling BFD on a specified IS-IS-enabled interface 196
Configuring BFD for IS-IS globally 197
BFD for VXLAN extension tunnels 197
Configuring BFD on a VXLAN extension tunnel 201
BFD for NSX tunnels 202

BFD for static routes 202
BFD considerations and limitations for static routes 203
BFD for static routes configuration 203
Configuring BFD on an IP static route 204
Configuring BFD on an IP static route in a nondefault VRF instance 205
Configuring BFD on an IPv6 static route 206
Configuring BFD on an IPv6 static route in a nondefault VRF instance 207
BFD for RSVP-TE LSP 208
BFD session creation for RSVP-TE LSP 209
BFD session deletion for RSVP-TE LSP 210
BFD session modification for RSVP-TE LSP 210
BFD session down handling for RSVP-TE LSP 210
BFD for RSVP-TE LSPs configuration 210
BFD session support per-router and per-interface module 211
BFD session down behavior 211
Enabling BFD for RSVP-TE LSPs at the global level 211
Enabling BFD for a specific RSVP-TE-LSP 212
Configuring BFD for the secondary path of an LSP 213
Enabling the IP router alert option 213
Displaying BFD information 214
Displaying BFD information 216
Displaying BFD information 220
Displaying BFD information 222
Displaying BFD information 225

BGP4 229
BGP4 overview 230
BGP support 231
Deployment scenarios 231
BGP4 overview 234
BGP4 peering 235
BGP4 message types 235
OPEN message 236
UPDATE message 236
NOTIFICATION message 237
KEEPALIVE message 237
REFRESH message 237
BGP4 attributes 238
BGP4 best path selection algorithm 238
BGP4 limitations and considerations 239
Implementation of BGP4 239
Device ID 240
BGP global mode 240
BGP global mode 241
Configuring a local AS number 242
IPv4 unicast address family 243
IPv4 multicast address family 245
Neighbor configuration 246
Configuring BGP4 neighbors 248
Requiring the first AS to be the neighbor AS 249
Peer groups 250

Configuring BGP4 peer groups 250
Advertising the default BGP4 route 251
Four-byte AS numbers 252
Enabling ASN capability globally 252
Enabling ASN capability for a BGP4 neighbor 253
Cooperative BGP4 route filtering 253
BGP4 parameters 254
Route redistribution 255
Redistributing routes into BGP4 255
Advertised networks 256
Importing routes into BGP4 257
Static networks 258
Configuring a static network 258
Route reflection 259
Configuring a cluster ID for a route reflector 259
Configuring a route reflector client 260
Route flap dampening 261
Aggregating routes advertised to BGP neighbors 261
Advertising the default BGP4 route 262
Advertising the default BGP4 route to a specific neighbor 263
Multipath load sharing 264
Specifying the weight added to received routes 264
Using the IPv4 default route as a valid next hop for a BGP4 route 265
Adjusting defaults to improve routing performance 266
Next-hop recursion 267
Enabling next-hop recursion 267
Enabling next-hop recursion in the IPv4 multicast address family 268
Route filtering 268
BGP VRF route filters 269
Import routes 269
Export routes 269
Regular-expression-based extended-community match filters 269
Using BGP VRF route filters 270
BGP regular expression pattern-matching characters 271
Timers 272
Enabling BGP4 in a non-default VRF 272
BGP4 outbound route filtering 273
Configuring BGP4 outbound route filtering 273
Enabling BGP4 cooperative route filtering 275
BGP4 confederations 276
Configuring BGP4 confederations 277
BGP community and extended community 278
Defining BGP4 extended communities 279
Applying a BGP4 extended community filter 280
BGP Large Communities 282
Introduction 282
BGP Large Communities attribute details 282
BGP Large Community configuration examples 283
BGP flowspec 285
Distribution of flowspec rules by BGP 286

Traffic filtering actions 287
Limitations 287
Configuration of BGP flowspec 289
Treatment of duplicate stanzas in a BGP flowspec route map 289
Enabling a BGP flowspec profile in TCAM 290
Configuring BGP flowspec rules 291
Enabling BGP flowspec IPv4 address family and activating neighbors 292
Distributing BGP flowspec rules 293
Updating a previously-distributed route map 294
Enabling statistics for BGP flowspec rules 296
Displaying and clearing statistics for IPv4 flowspec rules 297
Configuration example for BGP flowspec 298
BGP4 graceful restart 300
Configuring BGP4 graceful restart 300
BGP dynamic neighbors 303
Graceful restart for BGP dynamic neighbors 303
Configuring BGP dynamic neighbors 304
BGP add path overview 305
Advantages and limitations of BGP add path 306
BGP add path functionality 307
Negotiating BGP4 add paths capability 307
Advertising best BGP4 additional paths 308
Advertising all BGP4 additional paths 309
BGP additional-paths overview 310
Advantages of BGP additional-paths 310
Considerations and limitations for BGP additional-paths RIB-in 311
Considerations and limitations for BGP additional-paths RIB-out 311
Upgrade and downgrade considerations 311
BGP additional-paths functionality 311
Configuring BGP4 additional-paths and additional-path selection for the default VRF 312
Configuring BGP4 additional-paths and additional-path selection for a non-default VRF instance 313
Configuring BGP4 additional-paths for a specified neighbor 314
Configuring BGP additional-paths for a specified BGP4 neighbor for a non-default VRF instance 315
Disabling BGP additional-paths for a specified BGP4 neighbor 316
Configuring BGP additional-paths for a BGP peer group 316
BGP best external overview 318
Limitations of BGP best external 318
Upgrade and downgrade considerations 318
Configuring BGP4 best external 319
BGP additional-paths 319
Configuration of BGP additional-paths 321
Advantages of BGP additional-paths 321
Considerations and limitations for BGP additional-paths 322
Upgrade and downgrade considerations for BGP additional-paths 322
Configuring BGP additional-paths and additional-path selection at address-family level 323
Configuring BGP additional-paths and additional-path advertisement at neighbor level 324
Configuring additional-paths and advertisement of additional-paths at peer group level 325
Filtering additional-paths advertised at route-map level 326
Disabling BGP additional-paths capability for a neighbor 327
BGP best-external route 328

Configuring BGP best-external route at address-family level....329
Auto shutdown of BGP neighbors on initial configuration....330
Configuring auto shutdown of BGP neighbors on initial configuration....330
Disabling the BGP4 peer shutdown state....331
BGP4 graceful shutdown....332
Limitations....333
Configuring graceful shutdown for all BGP4 neighbors....334
Configuring graceful shutdown for a BGP4 peer group....335
Configuring graceful shutdown for a specific BGP4 neighbor....336
BGP automatic neighbor discovery....337
The role of sender....338
The role of receiver....338
BGP automatic neighbor discovery limitations....338
Static and automatic neighbor peer transition....339
Enabling BGP automatic neighbor discovery....339
Disabling the advertisement of TLV values for automatically discovered BGP neighbors....340
Disabling BGP auto neighbor discovery in LLDP at global level....341
Enabling the advertising of BGP neighbor TLV at interface level....341
Disabling BGP auto neighbor discovery in LLDP at interface level....342
Generalized TTL Security Mechanism support....343
Assumptions and limitations....343
Configuring GTSM for BGP4....344
Disabling the BGP AS_PATH check function....345
Using route maps....345
Matching on an AS-path....347
Matching on a community ACL....347
Matching on a destination network....348
Matching on a BGP4 static network....349
Matching on a next-hop device....351
Matching on an interface....351
Using route-map continue statements....352
Route-map continue statement for BGP4 routes....353
Using a route map to configure dampening....354
Clearing diagnostic buffers....354
Displaying BGP4 statistics....355
Displaying BGP4 statistics....357
Displaying BGP4 statistics....359
Displaying BGP4 statistics....361
Displaying BGP4 neighbor statistics....364
Displaying BGP4 neighbor statistics....367
Displaying BGP4 neighbor statistics....370
Clearing BGP4 dampened paths....374
Clearing BGP4 dampened paths....374
BGP PIC....375
Functional overview....375
Hierarchical RIB/FIB....376
BGP additional paths....376
Supported network triggers for failover....376
BGP PIC functional details....377
BGP PIC considerations and limitations....381

Configuring BGP PIC....381

BGP4+....383

BGP4+ overview....383
BGP global mode....384
IPv6 unicast address family....387
BGP4+ neighbors....390
Configuring BGP4+ neighbors using global IPv6 addresses....390
Configuring BGP4+ neighbors using link-local addresses....391
BGP4+ peer groups....393
Configuring BGP4+ peer groups....394
Configuring a peer group with IPv4 and IPv6 peers....395
Enabling ASN capability globally....397
Enabling ASN capability for a BGP4+ neighbor....397
Importing routes into BGP4+....398
Advertising the default BGP4+ route....399
Advertising the default BGP4+ route to a specific neighbor....399
Using the IPv6 default route as a valid next hop for a BGP4+ route....400
BGP4+ next hop recursion....401
Enabling next-hop recursion....402
BGP4+ NLRIs and next hop attributes....402
BGP4+ route reflection....403
Configuring a cluster ID for a route reflector....403
Configuring a route reflector client....404
BGP4+ route aggregation....405
Aggregating routes advertised to BGP neighbors....405
BGP4+ multipath....406
Enabling load-balancing across different paths....406
Route maps....408
Configuring a route map for BGP4+ prefixes....408
Redistributing prefixes into BGP4+....410
Redistributing routes into BGP4+....411
Specifying the weight added to BGP4+ received routes....411
Enabling BGP4+ in a non-default VRF....412
BGP4+ outbound route filtering....413
Configuring BGP4+ outbound route filtering....413
BGP4+ confederations....415
Configuring BGP4+ confederations....416
BGP4+ extended community....416
Defining BGP4+ extended communities....417
Applying a BGP4+ extended community filter....418
BGP4+ graceful restart....420
Configuring BGP4+ graceful restart....420
Auto shutdown of BGP neighbors on initial configuration....424
Configuring auto shutdown of BGP neighbors on initial configuration....424
Disabling the BGP4+ peer shutdown state....425
Generalized TTL Security Mechanism support....426
Assumptions and limitations....426
Configuring GTSM for BGP4+....427
Disabling the BGP AS_PATH check function....428
Displaying BGP4+ statistics....429

Displaying BGP4+ statistics....430
Displaying BGP4+ neighbor statistics....432
Displaying BGP4+ neighbor statistics....435
Clearing BGP4+ dampened paths....437
Clearing BGP4+ dampened paths....437

BGP EVPN for IP Fabrics....439
Overview....439
IP Fabrics topologies....439
IP Fabrics features....441
BGP EVPN use cases....442
BGP EVPN control plane....442
Supported route types....442
Supported BGP functionalities....442
Supported data plane encapsulation types....443
Supported service-interface model....443
BGP EVPN configuration examples....443
BGP EVPN neighbor examples....443
BGP EVPN instance configuration....446
BGP routing tables....449
BGP EVPN-based MCT cluster formation....450
Dependency on the EVPN instance....450
BGP EVPN peering for the MCT cluster....451
Handling of ES/AD routes in BGP....452
BGP EVPN-based VXLAN overlay....453
Underlay architectures....453
VXLAN overlay configuration....455
Dynamic VTEP discovery....456
MCT and logical VTEP....456
BGP EVPN in IPv6-only underlay....458
BGP EVPN in an MPLS provider network....458
EVPN next-hop resolution and tunnel-EVI membership....459
BGP EVPN in an MPLS provider network....460
Layer 2 (MAC) route exchange....462
MAC move detection and dampening....463
Automatic restoration of dampened routes....463
Aliasing in MPLS....463
Conversational MAC learning....465
Example output for "show mac-address-table"....465
ARP/ND (MACIP) route exchange....466
ARP suppression....466
Conversational ARP learning....467
EVPN prefix route exchange and Multi-VRF support....467
Symmetric or asymmetric routing....468
Import/export route-map filtering....468
Conversion of MACIP to host route to avoid traffic tromboning....469
Avoiding traffic tromboning in an MCT cluster....470
BGP EVPN VXLAN data center interconnect....471
Layer 2 and Layer 3 control-plane extension....471
Layer 2 handoff....471
Layer 3 handoff....471

EVPN Layer 3 interconnect 472
EVPN VPNv4/v6 interconnect 472
Static anycast gateway 473
IP unnumbered interface 475
High availability 476

OSPFv2 477
OSPFv2 overview 478
Autonomous System 478
OSPFv2 components and roles 479
Area Border Routers 479
Autonomous System Boundary Routers 479
Designated routers 479
Reduction of equivalent AS external LSAs 481
Algorithm for AS external LSA reduction 483
Enabling OSPFv2 483
Backbone area 483
Setting up the backbone area 484
Assigning OSPFv2 areas 485
Area range 485
Assigning an area range 486
Area types 486
Stub area and totally stubby area 487
Disabling summary LSAs for a stub area 487
Not-so-stubby area (NSSA) 488
Appendix-E implementation for OSPFv2 Type7 LSA 489
Configuring an NSSA 490
Configuring a summary-address for the NSSA 490
Configuring the translator role for a NSSA 491
Configuring the OSPF default metric 492
Preventing an NSSA ABR from generating external LSAs into a NSSA area 492
Assigning interfaces to an area 493
Assigning interfaces to an area 493
Link state advertisements 494
Link state advertisements 494
Configuring an MD5 password and authentication change hold time for an OSPFv2 interface 495
Virtual links 495
Configuring virtual links 496
Default route origination 498
External route summarization 499
SPF timers 499
Modifying Shortest Path First timers 500
OSPFv2 administrative distance 501
Configuring administrative distance based on route type 501
OSPFv2 LSA refreshes 502
Configuring the OSPFv2 LSA pacing interval 502
Support for OSPF RFC 2328 Appendix E 503
OSPFv2 graceful restart 504
Hitless upgrade support for OSPF graceful restart 505
Disabling OSPFv2 graceful restart 505
Re-enabling OSPFv2 graceful restart 506

Disabling OSPFv2 graceful restart helper ... 506
OSPFv2 stub router advertisement ... 507
OSPFv2 Shortest Path First throttling ... 508
IETF RFC and internet draft support ... 508
OSPFv2 non-stop routing ... 508
Limitations of NSR ... 509
BFD with OSPF NSR ... 509
Enabling OSPFv2 NSR ... 510
Synchronization of critical OSPFv2 elements ... 510
Link state database synchronization ... 511
LSA delayed acknowledging ... 511
LSA syncing and packing ... 511
Neighbor device synchronization ... 511
Synchronization limitations ... 511
Interface synchronization ... 512
Standby module operations ... 512
Neighbor database ... 512
LSA database ... 512
OSPFv2 distribute list ... 512
Configuring an OSPFv2 distribution list using ACLs ... 513
Configuring an OSPFv2 distribution list using route maps ... 514
OSPFv2 route redistribution ... 514
Redistributing routes into OSPFv2 ... 516
Load sharing ... 516
OSPFv2 type 3 LSA filtering ... 518
Usage and configuration guidelines ... 518
Configuring OSPFv2 type 3 LSA filtering ... 519
Verifying OSPFv2 type 3 LSA filtering configurations ... 520
Interface types to which the reference bandwidth does not apply ... 522
Changing the reference bandwidth for the cost on OSPFv2 interfaces ... 522
OSPFv2 over VRF ... 523
Enabling OSPFv2 in a non-default VRF ... 523
OSPF VRF-Lite for customer edge routers ... 524
OSPFv2 in a VCS environment ... 524
OSPFv2 considerations and limitations ... 525
Configuring the OSPFv2 Max-Metric Router LSA ... 526
Re-enabling OSPFv2 compatibility with RFC 1583 ... 527
Enabling OSPFv2 in a VCS environment ... 527
Changing default settings ... 529
Disabling and re-enabling OSPFv2 event logging ... 529
Understanding the effects of disabling OSPFv2 ... 530
Disabling OSPFv2 ... 530
Displaying OSPFv2 results ... 531
Supported scale for OSPFv2 and performance considerations ... 534
Displaying OSPFv2 results ... 535

OSPFv3 ... 537
OSPFv3 overview ... 537
OSPFv3 considerations and limitations ... 538
Configuring the router ID ... 538
Enabling OSPFv3 ... 538

Configuring OSPFv3....539
OSPFv3 areas....539
Backbone area....540
Area range....540
Area types....540
Assigning OSPFv3 areas....541
Assigning OSPFv3 areas to interfaces....542
Stub area and totally stubby area....544
Configuring a stub area....545
Not-so-stubby area....545
Configuring an NSSA....546
LSA types for OSPFv3....547
Virtual links....547
Virtual link source address assignment....549
Configuring virtual links....549
OSPFv3 route redistribution....551
Redistributing routes into OSPFv3....552
Default route origination....553
Configuring default external routes....554
Disabling and re-enabling OSPFv3 event logging....554
Disabling and re-enabling OSPFv3 event logging....555
Filtering OSPFv3 routes....555
Configuring an OSPFv3 distribution list using an IPv6 prefix list as input....556
Configuring an OSPFv3 distribution list using a route map as input....557
SPF timers....559
Modifying SPF timers....559
OSPFv3 administrative distance....560
Configuring administrative distance based on route type....560
Changing the reference bandwidth for the cost on OSPFv3 interfaces....561
OSPFv3 LSA refreshes....562
Configuring the OSPFv3 LSA pacing interval....563
External route summarization....563
OSPFv3 over VRF....564
Enabling OSPFv3 in a non-default VRF....564
Enabling OSPFv3 in a non-default VRF....565
Enabling OSPFv3 in a non-default VRF....566
Assigning OSPFv3 areas in a non-default VRF....566
Assigning OSPFv3 areas in a non-default VRF....567
Setting all OSPFv3 interfaces to the passive state....568
OSPFv3 graceful restart helper....569
Disabling OSPFv3 graceful restart helper....569
Re-enabling OSPFv3 graceful restart helper....570
OSPFv3 non-stop routing....571
Enabling OSPFv3 NSR....571
OSPFv3 max-metric router LSA....572
Configuring the OSPFv3 max-metric router LSA....572
IPsec for OSPFv3....573
IPsec for OSPFv3 configuration....575
IPsec for OSPFv3 considerations....575
Configuring IPsec on an OSPFv3 area....576

Configuring IPsec on an OSPFv3 interface....577
Configuring IPsec on OSPFv3 virtual links....578
Configuring IPsec on OSPFv3 virtual links....579
Specifying the key rollover and key add-remove timers....580
Clearing IPsec statistics....581
Displaying OSPFv3 results....582
Displaying OSPFv3 results....585
Displaying OSPFv3 results....588
Supported scale for OSPFv3 and performance considerations....593
Displaying OSPFv3 results....594
Clearing OSPFv3 redistributed routes....596

Multi-VRF....599
Multi-VRF overview....599
Configuring Multi-VRF....601
Configuring a VRF instance....601
Starting a routing process for a VRF....602
Assigning a Layer 3 interface to a VRF....602
Assigning a loopback interface to a VRF....603
Verifying a Multi-VRF configuration....603
Removing a VRF configuration....604
Configuring the maximum number of routes....604
Multi-VRF configuration example....605
Multi-VRF with eBGP and OSPF: Configuring PE1....606
Multi-VRF with eBGP and OSPF: Configuring PE2....610
Multi-VRF with eBGP and OSPF: Configuring CE1 and CE2....611
Multi-VRF with eBGP and OSPF: Configuring CE3 and CE4....611
Inter-VRF route leaking....612
Dynamic route-leak restrictions....612
Inter-VRF route conflicts....612
Displaying inter-VRF route leaking....613
Configuring static inter-VRF route leaking....614
Configuring dynamic inter-VRF route leaking....617
Internet route scaling....620
Configuring profile route....620

VRRPv2....623
VRRPv2 overview....623
VRRP terminology....628
VRRPv2 limitations on VDX devices....629
SLX-OS VRRP virtual router MAC address limitations....629
SLX-OS VRRP and VRRP-E interoperability....630
VRRP hold timer....630
VRRP interval timers....630
VRRP authentication....631
VRRP master device abdication to backup device....631
ARP and VRRP control packets....632
Enabling a master VRRP device....632
Enabling an owner VRRP device....636
Enabling a backup VRRP device....638
Enabling a backup VRRP device....639

Configuring simple text authentication on VRRP interfaces....640
Configuring MD5 authentication on VRRP interfaces....641
Abdicating VRRP master device status....642
VRRP multigroup clusters....643
Configuring multigroup VRRP routing....646
Tracked ports and track priority with VRRP and VRRP-E....650
Configuring VRRP port tracking....650
Tracking ports and setting the VRRP priority....654
VRRP owner preemption and interface tracking....655
Enabling VRRP owner preemption and interface tracking....655
VRRP backup preemption....656
Enabling VRRP backup preemption....657
Disabling VRRP backup preemption....658
Accept mode for backup VRRP devices....658
Disabling accept mode on a backup VRRP device....659
Accept mode for backup VRRP devices....659
Disabling accept mode on a backup VRRP device....659
Virtual router MAC address....660
Configuring unique virtual MAC addresses per VRID....660
Configuring unique virtual MAC addresses per VRID....662
Suppressing RIP route advertisements on VRRP backup devices....663
VRRP-Ev2 overview....664
Enabling a VRRP-E device....664
Enabling a VRRP-E device....666
Configuring MD5 authentication on IPv4 VRRP-E interfaces....667
Track routes and track priority with VRRP-E....669
Configuring VRRP-E route tracking....669
VRRP-E load-balancing using short-path forwarding....672
Packet routing with short-path forwarding to balance traffic load....672
Short-path forwarding with revert priority....673
Configuring VRRP-E load-balancing using short-path forwarding....674
Configuring VRRP-E load-balancing using short-path forwarding....675
VRRP-E slow start timer....676
Configuring a VRRP-E slow-start timer....676
Multiple virtual IP address support for VRRP-E....677
Configuring multiple virtual IP addresses for VRRP-E....677
Displaying multiple virtual IP addresses for VRRP-E information....679
VRRP-E scaling using logical groups....680
Configuring VRRP-E scaling....681
Displaying VRRP-E scaling information....682
Displaying VRRPv2 information....682
Clearing VRRPv2 statistics....687

VRRPv3....691
VRRPv3 overview....691
VRRPv3 functionality differences on VDX devices....692
VRRPv3 performance and scalability metrics for Network OS devices....692
Enabling IPv6 VRRPv3....692
Enabling an IPv6 VRRPv3 owner device....694
Enabling an IPv6 VRRPv3 backup device....695
Enabling IPv4 VRRPv3....696

Enabling an IPv4 VRRPv3 owner device....698
Enabling an IPv4 VRRPv3 backup device....699
Tracked ports and track priority with VRRP and VRRP-E....700
Port tracking using IPv6 VRRPv3....700
Tracking ports and setting VRRP priority using VRRPv3....702
Tracked IPsec tunnels and track priority with VRRP and VRRP-E....703
Configuring VRRP tracking for IPsec tunnels....704
Configuring VRRP-E tracking for IPsec tunnels....705
VRRP hold timer....706
Configuring VRRP hold timer support....706
Accept mode for backup VRRP devices....708
Disabling accept mode on a backup VRRP device....708
Accept mode for backup VRRP devices....709
Enabling accept mode on a backup VRRP device....709
Accept mode for backup VRRP devices....711
Disabling accept mode on a backup VRRP device....711
Alternate VRRPv2 checksum for VRRPv3 IPv4 sessions....711
Enabling the v2 checksum computation method in a VRRPv3 IPv4 session....712
Enabling the VRRPv2 checksum computation method in a VRRPv3 IPv4 session....713
Displaying alternate VRRPv2 checksum settings....714
VRRPv3 router advertisement suppression....714
Disabling VRRPv3 router advertisements....715
Automatic generation of a virtual link-local address for VRRPv3....716
Enabling auto-generation of an IPv6 virtual link-local address....716
Displaying VRRPv3 statistics....718
Clearing VRRPv3 statistics....721
VRRP-Ev3 Overview....724
Enabling IPv6 VRRP-Ev3....724
Enabling an IPv6 VRRP-Ev3 device....725
Configuring MD5 authentication on IPv6 VRRP-Ev3 interfaces....727
VRRP-E load-balancing using short-path forwarding....728
Packet routing with short-path forwarding to balance traffic load....728
Short-path forwarding with revert priority....729
Configuring VRRP-Ev3 load-balancing....730
VRRP-Ev3 sub-second failover....731
Configuring sub-second failover using VRRP-Ev3....731
Displaying and clearing VRRP-Ev3 statistics....732

VXLAN....737
VXLAN Transit Mode....737
UDA in Transit mode....737
VXLAN Native IP Mode....738
UDA in Native IP mode....738

Preface

- Conventions....21
- Documentation and Training....22
- Getting Help....22
- Providing Feedback....23

This section describes the text conventions used in this document, where you can find additional information, and how you can provide feedback to us.

Conventions

This section discusses the conventions used in this guide.

Notes, cautions, and warnings

Notes, cautions, and warning statements may be used in this document. They are listed in the order of increasing severity of potential hazards.

NOTE
A Note provides a tip, guidance, or advice, emphasizes important information, or provides a reference to related information.

ATTENTION
An Attention statement indicates a stronger note, for example, to alert you when traffic might be interrupted or the device might reboot.

CAUTION
A Caution statement alerts you to situations that can be potentially hazardous to you or cause damage to hardware, firmware, software, or data.

DANGER
A Danger statement indicates conditions or situations that can be potentially lethal or extremely hazardous to you. Safety labels are also attached directly to products to warn of these conditions or situations.

Text formatting conventions

Text formatting conventions such as boldface, italic, or Courier font may be used to highlight specific words or phrases.

Format	Description
bold text	Identifies command names.
	Identifies keywords and operands.
	Identifies the names of GUI elements.
	Identifies text to enter in the GUI.
italic text	Identifies emphasis.
	Identifies variables.
	Identifies document titles.

Format	Description
`Courier font`	Identifies CLI output. Identifies command syntax examples.

Command syntax conventions

Bold and italic text identify command syntax components. Delimiters and operators define groupings of parameters and their logical relationships.

Convention	Description
bold text	Identifies command names, keywords, and command options.
italic text	Identifies a variable.
[ ]	Syntax components displayed within square brackets are optional. Default responses to system prompts are enclosed in square brackets.
{ **x** | **y** | **z** }	A choice of required parameters is enclosed in curly brackets separated by vertical bars. You must select one of the options.
x | **y**	A vertical bar separates mutually exclusive elements.
< >	Nonprinting characters, for example, passwords, are enclosed in angle brackets.
...	Repeat the previous element, for example, *member*[*member*...].
\	Indicates a "soft" line break in command examples. If a backslash separates two lines of a command input, enter the entire command at the prompt without the backslash.

Documentation and Training

Find Extreme Networks product information at the following locations:

- Current Product Documentation
- Release Notes
- Hardware/software compatibility matrices for Campus and Edge products
- Supported transceivers and cables for Data Center products
- Other resources, like white papers, data sheets, and case studies

Extreme Networks offers product training courses, both online and in person, as well as specialized certifications. For details, visit www.extremenetworks.com/education/.

Getting Help

If you require assistance, contact Extreme Networks using one of the following methods:

Extreme Portal	Search the GTAC (Global Technical Assistance Center) knowledge base; manage support cases and service contracts; download software; and obtain product licensing, training, and certifications.
The Hub	A forum for Extreme Networks customers to connect with one another, answer questions, and share ideas and feedback. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.
Call GTAC	For immediate support: (800) 998 2408 (toll-free in U.S. and Canada) or 1 (408) 579 2826. For the support phone number in your country, visit: www.extremenetworks.com/support/contact

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number, or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any actions already taken to resolve the problem
- A description of your network environment (such as layout, cable type, other relevant environmental information)
- Network load at the time of trouble (if known)
- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any related RMA (Return Material Authorization) numbers

Subscribe to Service Notifications

You can subscribe to email notifications for product and software release announcements, Vulnerability Notices, and Service Notifications.

1. Go to www.extremenetworks.com/support/service-notification-form.
2. Complete the form (all fields are required).
3. Select the products for which you would like to receive notifications.

   NOTE
   You can modify your product selections or unsubscribe at any time.

4. Select **Submit**.

Providing Feedback

The Information Development team at Extreme Networks has made every effort to ensure the accuracy and completeness of this document. We are always striving to improve our documentation and help you work better, so we want to hear from you. We welcome all feedback, but we especially want to know about:

- Content errors, or confusing or conflicting information.
- Improvements that would help you find relevant information in the document.
- Broken links or usability issues.

If you would like to provide feedback, you can do so in three ways:

- In a web browser, select the feedback icon and complete the online feedback form.
- Access the feedback form at https://www.extremenetworks.com/documentation-feedback/.
- Email us at documentation@extremenetworks.com.

Provide the publication title, part number, and as much detail as possible, including the topic heading and page number if applicable, as well as your suggestions for improvement.

About This Document

- Supported hardware and software.....25
- What's new in this document.....25

Supported hardware and software

In those instances in which procedures or parts of procedures documented here apply to some devices but not to others, this guide identifies exactly which devices are supported and which are not.

Although many different software and hardware configurations are tested and supported by Extreme Networks, Inc. for this SLX-OS release, documenting all possible configurations and scenarios is beyond the scope of this document.

The following hardware platforms are supported by this release:

- ExtremeSwitching SLX 9140
- ExtremeSwitching SLX 9240

NOTE
Some of the commands in this document use a slot/port designation. Because the SLX 9140 and the SLX 9240 do not contain line cards, the slot designation must always be "0" (for example, 0/1 for port 1).

What's new in this document

The following table describes information added to this guide for the SLX-OS 18s.1.03a software release.

TABLE 1 Summary of enhancements

Feature	Description	Described in
VXLAN Transit mode	New topic for the guide.	VXLAN Transit Mode on page 737
VXLAN Native IP mode	New topic for the guide.	VXLAN Native IP Mode on page 738

ARP

- ARP and Neighbor Discovery overview....27
- Basic ARP configuration....27
- Dynamic ARP Inspection (DAI)....32
- ARP Guard....38
- ARP and ND suppression....42
- Conversational ARP and ND....45

ARP and Neighbor Discovery overview

When forwarding traffic, a device needs to know the destination's MAC address, because each IP packet is encapsulated in an Ethernet frame. The MAC address is needed not only for the packet's final destination but also for a next hop towards the destination.

The technology by which a device gets the MAC address varies between IPv4 and IPv6, as follows:

- IPv4: Address Resolution Protocol (ARP)
- IPv6: Neighbor Discovery (ND). For basic ND configuration, refer to "IPv6 neighbor discovery configuration."

Basic ARP configuration

The Address Resolution Protocol (ARP) maps IPv4 network addresses to MAC hardware addresses.

When forwarding traffic, a device needs to know the destination MAC address, because each IP packet is encapsulated in an Ethernet frame. MAC addresses are needed not only for the packet's final destination but also for a next hop towards the destination. A device first searches its ARP cache; a match for the IP address supplies the corresponding MAC address. Otherwise, the device broadcasts an ARP request. The network devices receive such ARP requests, and the host with a matching IP address sends an ARP reply that includes its MAC address.

After a matching ARP reply is received by the device, the following events occur:

- The packet is sent towards its destination.
- The IP address/MAC address pair is added to the ARP cache, as a dynamic ARP entry.

An aging timer—by default 25 minutes—is triggered when a dynamic entry is added to the ARP cache and is reset if an ARP reply is received. The aging timer ensures that the ARP cache does not retain learned entries that are no longer valid. An entry can become invalid when the device with the MAC address of the entry is no longer on the network.

You can also add static ARP entries to the cache. Static entries do not time out. They are useful to preconfigure an entry for a device not yet connected or to prevent an entry from aging out.

Modifying the ARP aging timeout

The ARP aging timeout configured for an interface overrides the global aging timeout (25 minutes).

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Access the interface on which you are modifying the ARP aging timeout.

```
device(config)# interface ethernet 2/1
```

```
device(config)# interface ethernet 0/1
```

3. Enter the **ip arp-aging-timeout** command, specifying the new value.

```
device(conf-if-eth-2/1)# ip arp-aging-timeout 100
```

```
device(conf-if-eth-0/1)# ip arp-aging-timeout 100
```

   Values range from 0 through 240 minutes.

The following example changes an interface ARP aging timeout to 100 minutes.

```
device# configure terminal
device(config)# interface ethernet 2/1
device(conf-if-eth-2/1)# ip arp-aging-timeout 100

device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# ip arp-aging-timeout 100
```

Enabling and disabling ARP learning

Use this procedure to enable ARP learning not only locally, but from all ARP requests. ARP learning decreases the time needed to populate the ARP cache.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command to access RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **interface ve** command to access VE configuration mode.

```
device(config-rbridge-id-1)# interface ve 110
```

```
device(config)# interface ve 110
```

4. To enable fabric learning, enter the **ip arp learn-any** command.

```
device(config-ve-110)# ip arp learn-any
```

   To disable fabric learning, enter the **no ip arp learn-any** command.

Creating static ARP entries

Static ARP entries are useful to preconfigure an entry for a device not yet connected or to prevent an entry from aging out.

On a NOS device, IPv4 fabric-virtual-gateway active sessions must be enabled to respond to unicast ARP requests. The following example shows how to configure the gateway MAC address for an IPv4 Fabric-Virtual-Gateway session to respond to unicast ARP requests.

```
device(config)# router fabric-virtual-gateway
device(conf-router-fabric-virtual-gateway)# address-family ipv4
device(conf-address-family-ipv4)# accept-unicast-arp-request
```

NOTE
You can define a static ARP entry for a physical interface or for a VE. The flow in this topic is for a physical interface.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **arp** command, specifying the IP address, the MAC address, and the interface.

```
device(config)# arp 10.53.4.2 1245.7654.2348 interface ethernet 1/2
```

The example creates a static ARP entry for IP address 10.53.4.2 and associates it with MAC address 1245.7654.2348 and physical port 1/2.

```
device(config)# arp 10.53.4.2 1245.7654.2348 interface ethernet 0/2
```

The example creates a static ARP entry for IP address 10.53.4.2 and associates it with MAC address 1245.7654.2348 and physical port 0/2.

3. Enter the keyword **arp** followed by the IP address and the MAC address. As an option, on the same line enter the interface or VLAN associated with the entry.

```
device(config)# arp 10.53.4.2 1245.7654.2348 vlan 10
```

The example creates a static ARP entry for IP address 10.53.4.2 and associates it with MAC address 1245.7654.2348 and VLAN 10.

4. Enter interface configuration mode.

```
device(config)# interface tengigabitethernet 2/0/33
```

5. Turn on channel-group 20 mode for the interface.

```
device(conf-if-te-2/0/33)# channel-group 20 mode on
device(conf-if-te-2/0/33)# exit
```

6. From global configuration mode, enter rbridge configuration mode for the rbridge-id where the static ARP entry is to be created.

```
device(config)# rbridge-id 2
```

7. Enter the **arp** command followed by the IP address and the MAC address and the port-channel group for the interface.

```
sw0(config-rbridge-id-2)# arp 2.2.2.2 1232.1211.1333 interface Port-channel 20
```

The example creates a static ARP entry for Rbridge-id 2. The entry is for IP address 2.2.2.2 and MAC address 1232.1211.13. The entry is associated with the interface on port-channel 20.

The following example creates a basic static ARP entry for IP address 10.1.1.2 and MAC address 0003.0004.0005.

```
device# configure terminal
 device(config)# arp 10.1.1.2 0003.0004.0005
```

The following example creates a static ARP entry that associates an IP address, a MAC address, and a physical port.

```
device# configure terminal
device(config)# arp 10.53.4.2 1245.7654.2348 ethernet 2/1
```

The following example configures the 2/0/33 interface as a member of port-channel 20 and then defines a static ARP for that port channel.

```
device# configure terminal
device(config)# interface tengigabitethernet 2/0/33
device(conf-if-te-2/0/33)# channel-group 20 mode on
device(conf-if-te-2/0/33)# exit
device(config)# rbridge-id 2
sw0(config-rbridge-id-2)# arp 2.2.2.2 1232.1211.1333 interface Port-channel 20
```

Creating a VRF static ARP entry

In a VRF instance, static ARP entries are useful to preconfigure an entry for a device not yet connected or to prevent an entry from aging out.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **vrf** command to access VRF configuration mode.

   ```
   device(config)# vrf test
   ```

3. Enter the **address-family** command, specifying the needed parameters.

   ```
   device(config-vrf-test)# address-family ipv4 unicast
   ```

4. Enter the **arp** command, specifying the IP address, the MAC address, and the interface.

   ```
   device(vrf-test-ipv4-unicast)# arp 10.6.6.7 0001.0001.0001 interface ethernet 2/1
   ```

   ```
   device(vrf-test-ipv4-unicast)# arp 10.6.6.7 0001.0001.0001 interface ethernet 0/1
   ```

The following example accesses a VRF instance, specifies the address family, and defines a static ARP.

```
device# configure terminal
device(config)# vrf test
device(config-vrf-test)# address-family ipv4 unicast
device(vrf-test-ipv4-unicast)# arp 10.6.6.7 0001.0001.0001 interface ethernet 2/1

device# configure terminal
device(config)# vrf test
device(config-vrf-test)# address-family ipv4 unicast
device(vrf-test-ipv4-unicast)# arp 10.6.6.7 0001.0001.0001 interface ethernet 0/1
```

Proxy ARP

Proxy ARP allows a device to answer ARP requests from devices in one network on behalf of devices in another network.

Because ARP requests are MAC-layer broadcasts, they reach only the devices that are directly connected to the sender of the ARP request. ARP requests do not cross routers.

However, if Proxy ARP is enabled on a device connected to two subnets, the device can respond to an ARP request from the other subnet. For example, if it is connected to the two subnets 10.10.10.0/24 and 10.20.20.0/24, the device can respond to an ARP request from 10.10.10.69 for the MAC address of the device with the IP address 10.20.20.69. In standard ARP, a request from a device in the 10.10.10.0/24 subnet cannot reach a device in the 10.20.20.0 subnet. Consequently, the request is not answered.

The ARP reply returned contains the device's MAC address instead of the MAC address of the target host. In this transaction, the traffic sent to the target host is forwarded through Layer 3 rather than being switched through Layer 2.

NOTE
Under some Layer 2 configurations such as an uplink switch or private VLAN, broadcast packets are not flooded to every port in a VLAN. In these configurations, an ARP request from one host may not reach another host. Enabling Proxy ARP locally on a port directs the device to reply on behalf of a target host if it exists.

Enabling Proxy ARP on an interface

Complete the following steps to enable Proxy ARP on a physical or VE interface.

NOTE
Proxy ARP is disabled by default.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Access the interface on which you are enabling Proxy ARP.

```
device(config)# interface ethernet 2/1
device(config)# interface ethernet 0/1
```

3. Enter the **ip proxy-arp** command.

```
device(conf-if-eth-2/1)# ip proxy-arp
device(conf-if-eth-0/1)# ip proxy-arp
```

The following example enables Proxy ARP.

```
device# configure terminal
device(config)# interface ethernet 2/1
device(conf-if-eth-2/1)# ip proxy-arp

device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# ip proxy-arp
```

ARP show and clear commands

There are ARP show and clear commands, listed here with descriptions.

TABLE 2 ARP show commands in the *SLX-OS Command Reference*

Command	Description
show arp	Displays the Address Resolution Protocol (ARP) entries. You can filter the display by interface or VRF. You can also display only dynamic or only static ARP entries.

TABLE 3 ARP clear commands in the *SLX-OS Command Reference*

Command	Description
clear arp	The **clear arp** option clears ARP entries, triggering ARP requests for the cleared entries. The **clear arp no-refresh** option clears ARP entries without such requests. You can limit the clearing by interface or VRF. You can also clear for a specified next-hop IP address.

Dynamic ARP Inspection (DAI)

Dynamic ARP Inspection (DAI) is a security feature that validates address resolution protocol (ARP) packets in a subnet, and discards packets with invalid IP address and MAC address bindings.

ARP poisoning

An ARP poisoning attack, also known as ARP spoofing, targets the ARP caches of devices connected to the subnet, with the goal of intercepting traffic. A malicious host might use one of the following tactics:

- Send ARP packets claiming to have an IP address that actually belongs to another host.
- Reply to an ARP request with its own MAC address, thereby causing other hosts on the subnet to store this information in their ARP tables, even replacing an existing ARP entry.
- Send gratuitous replies without having received any ARP requests.

If the poisoning succeeds, traffic intended for the device under attack is instead routed to the attacker computer. The attacker has various options:

- Not forward any traffic to the computer under attack or forward some of the traffic, but not all of it (denial-of-service attacks).
- Forward inspected traffic to the compromised device (interception).
- Modify the traffic and then forward it (man-in-the-middle attack).

The current release supports two features that protect against ARP poisoning:

- DAI, described in the current section
- ARP Guard, described in ARP Guard on page 38

The following table compares DAI and ARP Guard for protection against ARP poisoning:

TABLE 4 ARP Guard and DAI

Aspect	DAI	ARP Guard
Flow-based	No. Applies to all VLAN ARP packets.	Flow-based, which can prevent high CPU load.
Per port	No. Applies to all VLAN ports.	Applied per port or VPLS end-point.
Rate-limiting	No rate-limiting option.	Rate limiting is supported.
TCAM load	Low TCAM load.	Medium TCAM load.

DAI functionality

> **NOTE**
> DAI is currently supported only in non-DHCP environments.

On VLANs, Dynamic ARP Inspection (DAI) can examine incoming ARP packets. DAI discards packets with invalid IP/MAC address bindings, guarding against ARP-poisoning attacks; only valid ARP requests and responses are relayed. You specify valid, static IP/MAC address bindings in the **permit** statements of ARP ACLs.

Towards enabling DAI, you need to decide which ports you are defining as trusted and which as untrusted. ARP packets on trusted ports bypass all DAI validations and are forwarded as required. DAI examines ARP packets only on untrusted ports.

When DAI is implemented on a VLAN, it monitors untrusted ports as follows:

- Intercepts ARP requests and responses
- Compares the IP/MAC address bindings with the **permit** statements in the ACL applied to the VLAN
- Drops invalid packets
- Forwards valid packets to the appropriate destination, with the option of generating log entries
- Forwards valid packets to the appropriate destination

The following table summarizes DAI enablement/disablement on trusted/untrusted ports:

TABLE 5 DAI on trusted and untrusted ports

VLAN setting	Port setting	Action
DAI disabled	Trusted or untrusted	All incoming ARP packets are hardware-forwarded.
DAI enabled	Trusted	All incoming ARP packets are trapped to the CPU and then software-forwarded.
DAI enabled	Untrusted	All incoming ARP packets are trapped to the CPU. Following DAI, the packets are software-forwarded or dropped.

DAI implementation

Address Resolution Protocol (ARP) access control lists (ACLs), applied to untrusted ports, permit only ARP packets with specified IP/MAC address bindings. Such ACLs implement Dynamic ARP Inspection (DAI).

DAI configuration guidelines

Follow these guidelines when implementing ARP ACLs for DAI.

- DAI is available on Layer 2 VLANs and Layer 3 virtual Ethernets (VE).

  NOTE
  In these guidelines, the term VLAN can also include VEs (where relevant).

- DAI is available on Layer 2 VLANs.
- VLANs supported for DAI include:
  - 802.1Q VLANs
  - VLANs that cross multiple VCS clusters
  - Virtual fabric (VFAB) VLANs
  - VE interfaces under virtual routing and forwarding (VRF). Both default and non-default VRFs are supported.
- DAI is not supported for management interfaces or Layer 3 router interfaces.
- DAI is not supported for management interfaces.
- DAI is not supported on a VLAN to which a VE is bound.
- On a VLAN with DAI enabled, the following types of member ports are supported for DAI:
  - Physical interfaces (in switchport mode)
  - Port-channel interfaces (LAGs or vLAGs) (in switchport mode)
  - Port-channel interfaces (LAGs or MLAGs) (in switchport mode)
  - Port-channel interfaces (LAGs or MLAGs) (in switchport mode)

- DAI on bridge-domain is not currently supported.
- In a VCS VLAN with multiple RBridges, enabling DAI on the primary RBridge automatically implements it on all of the VLAN RBridges.
- On DAI-enabled VLANs, the rate limit per chip is 64 ARP packets per second (pps).

Creating an ARP ACL

Use this procedure to create an ARP ACL and **permit ip host** rules.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **arp access-list** command to create the ACL.

```
device(config)# arp access-list ARP_ACL_01
```

3. For each ACL rule, enter a **permit ip host** command.

```
device(config-arp-acl)# permit ip host 1.1.1.1 mac host 0020.2222.2222
device(config-arp-acl)# permit ip host 1.1.1.2 mac host 0020.2222.2223
```

Applying an ARP ACL to a VLAN

Use this procedure to apply an ARP ACL to a VLAN.

NOTE
To replace an ARP ACL on a VLAN, there is no need to remove a previously applied ACL. The most recent ACL applied replaces any previous ACL.

1. Enter **configure terminal** to change to global configuration mode.

```
device# configure terminal
```

2. Enter the **vlan** command to access the VLAN.

```
device(config)# vlan 200
```

3. Enter the **ip arp inspection filter** command, specifying the ACL.

```
device(config-vlan-200)# ip arp inspection filter ARP_ACL_01
```

4. To return to global configuration mode, (for example, to define trusted and untrusted interfaces and to enable DAI), enter **exit**.

```
device(config-vlan-200)# exit
```

Defining trusted and untrusted interfaces under DAI

Use this procedure to specify untrusted and trusted interfaces under Dynamic ARP Inspection (DAI).

1. Enter **configure terminal** to change to global configuration mode.

```
device# configure terminal
```

2. For each interface that you need to define as trusted or untrusted, complete the following steps:
   a) Enter the **interface** command to access interface configuration mode.

```
device(config)# interface tengigabitethernet 1/0/1
device(config)# interface ethernet 2/3
device(config)# interface ethernet 0/3
```

   b) To define the interface as trusted, enter the **ip arp inspection trust** command.

```
device(config-if-te-1/0/1)# ip arp inspection trust
device(conf-if-eth-2/3)# ip arp inspection trust
device(conf-if-eth-0/3)# ip arp inspection trust
```

   c) To redefine a currently trusted interface as untrusted (default), enter the **no ip arp inspection trust** command.

```
device(config-if-te-1/0/1)# no ip arp inspection trust
device(conf-if-eth-2/3)# no ip arp inspection trust
device(conf-if-eth-0/3)# no ip arp inspection trust
```

The following example defines a port-channel interface as trusted.

```
device# configure terminal
device(config)# interface port-channel 200
device(config-Port-channel-200)# ip arp inspection trust
```

Enabling and disabling DAI

Use this procedure to enable Dynamic ARP Inspection (DAI) on a VLAN.

1. Enter **configure terminal** to change to global configuration mode.

```
device# configure terminal
```

2. Enter the **interface vlan** command to access configuration mode on the VLAN for which you are enabling DAI.

```
device(config)# interface vlan 1001
```

3. Enter the **vlan** command to access configuration mode on the VLAN for which you are enabling DAI.

```
device(config)# vlan 1001
```

4. To enable DAI, enter the **ip arp inspection** command.

```
device(config-if-vlan-1001)# ip arp inspection
device(config-vlan-1001)# ip arp inspection
```

5. To disable DAI, enter the **no ip arp inspection** command.

```
device(config-if-vlan-1001)# no ip arp inspection
device(config-vlan-1001)# no ip arp inspection
```

Enabling and disabling DAI logging

Use this procedure to enable Dynamic ARP Inspection (DAI) on a VLAN, with terminal logging.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **arp access-list** command to create the access list.

```
device(config)# arp access-list arpACL1
```

3. For each ACL rule, enter a **permit ip host** command.

```
device(config-arp-acl)# permit ip host 1.1.1.1 mac host 0020.2222.2222 log
device(config-arp-acl)# permit ip host 1.1.1.2 mac host 0020.2222.2223
```

   NOTE
   If there is no permit statement that contains the **log** keyword, DAI logging does not occur.

4. Enter the **exit** command to return to global configuration mode.

```
device(config-arp-acl)# exit
```

5. Enter the **interface vlan** command to access configuration mode on the VLAN for which you are enabling DAI.

```
device(config)# interface vlan 200
```

6. Enter the **vlan** command to access configuration mode on the VLAN for which you are enabling DAI.

```
device(config)# vlan 200
```

7. Enter the **ip arp inspection filter** command, specifying an ACL containing a **permit** statement with the **log** keyword.

```
device(config-vlan-200)# ip arp inspection filter arpACL1
```

8. Enter the **ip arp inspection logging acl-match matchlog** command.

```
device(config-vlan-200)# ip arp inspection logging acl-match matchlog
```

9. Enter the **ip arp inspection** command to enable DAI.

```
device(config-vlan-200)# ip arp inspection
```

10. Enter the **end** command to return to privileged EXEC mode.

```
device(config-vlan-200)# end
```

11. Enter the terminal monitor command.

```
device# terminal monitor
```

The following example applies ARP_ACL_01 to VLAN 200, enables DAI logging on VLAN 200, enables DAI, and displays the log.

```
device# configure terminal
device(conf)# interface vlan 200
device(config-vlan-200)# ip arp inspection filter ARP_ACL_01
device(config-vlan-200)# ip arp inspection logging acl-match
device(config-vlan-200)# ip arp inspection
device(config-vlan-200)# end
device# terminal monitor
Terminal monitoring is enabled.
device# 015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11

device# configure terminal
device(conf)# vlan 200
device(conf-vlan-200)# ip arp inspection filter ARP_ACL_01
device(conf-vlan-200)# ip arp inspection logging acl-match
device(conf-vlan-200)# ip arp inspection
device(conf-vlan-200)# end
device# terminal monitor
Terminal monitoring is enabled.
device# 015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
```

DAI show and clear commands

There is a full range of Dynamic ARP Inspection (DAI) show and clear commands, listed here with descriptions.

TABLE 6 DAI show commands in the *SLX-OS Command Reference*

Command	Description
show arp access-list	For one or all ARP ACLs defined on a device, displays ACL names and their **permit** statements.
show ip arp inspection interfaces	For VLANs enabled for DAI, displays a list of trusted interfaces.

TABLE 6 DAI show commands in the *SLX-OS Command Reference* (continued)

Command	Description
show ip arp inspection statistics	Displays DAI statistics for one or more DAI-enabled VLANs.
show ip arp inspection	Displays DAI information for one or more VLANs.

TABLE 7 DAI clear commands in the *SLX-OS Command Reference*

Command	Description
clear ip arp inspection statistics	Clears DAI statistics for all DAI-enabled VLANs.

ARP Guard

ARP Guard is an alternative to Dynamic ARP Inspection (DAI) for protection against ARP poisoning.

Internet exchange points (IXPs) are designed based on a flat Layer 2 topology to provide any-to-any connectivity among BGP routers from different ISPs, CSPs, and enterprises connected to it. As an IP host, each BGP peering router makes use of ARP protocol to determine the MAC address of its BGP peers. Since ARP is not a secure protocol, any BGP router can reply to the ARP request for any IP address, and any BGP router can generate gratuitous ARP to claim ownership of any IP address in the router. When the network administrator of a BGP border router connecting to the IXP incorrectly configures the router IP address or unknowingly turns on the ARP Proxy feature on the interface facing the IXP, valid traffic may be sent to the wrong destination.

ARP Guard, like DAI, is also effective against the various methods of ARP poisoning. For details, refer to Dynamic ARP Inspection (DAI) on page 32.

The ARP Guard feature uses a set of ACL-like commands to build a table of allowed IP addresses on the link. As a result, when an ARP reply—either due to gratuitous ARP or in response to a normal ARP request—is received on a port facing the BGP router, the ARP reply is inspected based on the IP address parameters configured for **permit** commands. ARP packets that do not match the entries in the ACL are dropped. Matching ARP packets are forwarded.

ARP Guard and DAI

The following table compares ARP Guard and Dynamic ARP Inspection (DAI) for protection against ARP poisoning:

TABLE 8 Aspects of ARP Guard and DAI

Aspect	DAI	ARP Guard
Flow-based	No. Applies to all VLAN ARP packets.	Flow-based, which can prevent high CPU load.
Per port	Applies to all untrusted ports on a VLAN.	Applied per port or VPLS endpoint.
Rate limiting	No rate limiting option.	Rate limiting is supported.
TCAM load	Low TCAM load.	Medium TCAM load.

ARP Guard implementation

Implementing ARP Guard requires creating ARP ACLs, then applying and enabling them on interfaces and port-channels.

ARP Guard configuration guidelines

Follow these guidelines when implementing ARP Guard:

- ARP Guard is supported on physical interfaces and port-channels.
- ARP Guard is not supported on Layer 3 interfaces.
- ARP Guard requires that the **switchport** command be implemented on the interface.
- If ARP Guard is enabled on a port-channel and a member port is removed, ARP Guard properties are retained on the removed port.
- On an interface where ARP Guard is enabled, you must limit the rate of ARP traffic by a Layer 2 ACL.

The following table indicates which hardware profiles support various type of ARP Guard implementation.

> NOTE
> For descriptions of the hardware profiles, refer to **show hardware profile** in *Extreme SLX-OS Command Reference*.

TABLE 9 Profile Support for ARP Guard

Profile	Basic ARP Guard	ARP Guard with Rate Limiting	ARP Guard on VPLS Endpoint	ARP Guard on VPLS End-point with Rate Limiting
default	Yes	No	No	No
layer2-optimised-1	Yes	Yes	Yes	Yes
npb-optimised-1	Yes	No	No	No
openflow-optimised-1	Yes	No	No	No
openflow-optimised-2	Yes	Yes	No	No
openflow-optimised-3	Yes	Yes	No	No
vxlan-ext	Yes	No	No	No

Creating an ARP ACL

Use this procedure to create an ARP ACL and **permit ip host** rules.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **arp access-list** command to create the ACL.

```
device(config)# arp access-list ARP_ACL_01
```

3. For each ACL rule, enter a **permit ip host** command.

```
device(config-arp-acl)# permit ip host 1.1.1.1 mac host 0020.2222.2222
device(config-arp-acl)# permit ip host 1.1.1.2 mac host 0020.2222.2223
```

Applying an ARP ACL to an interface or port-channel

For ARP Guard, use this procedure to apply an ARP ACL to an interface or port-channel.

NOTE
To replace an ARP ACL, there is no need to remove a previously applied ACL. The most recent ACL applied replaces any previous ACL.

1. Enter **configure terminal** to change to global configuration mode.

```
device# configure terminal
```

2. Enter the **interface** command.

```
device(conf)# interface ethernet 1/2
```

3. Enter the **switchport** command.

```
device(conf-if-eth-1/2)# switchport
```

4. Enter the **ip arp inspection filter** command, specifying the ACL.

```
device(conf-if-eth-1/2)# ip arp inspection filter ARP_ACL_01
```

5. Enter **exit** to return to global configuration mode.

```
device(conf-if-eth-1/2)# exit
device(conf)#
```

Enabling ARP Guard on an interface or port-channel

Use this procedure to enable ARP Guard on a physical interface or a port-channel.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **mac access-list extended** command to create an ACL to enable ARP Guard.

```
device(config)# mac access-list extended arp_guard_enable_1
```

3. In the enabling ACL, create appropriate rules.

```
device(conf-macl-ext)# permit host 0014.2211.1111 any vlan 100 arp arp-guard
device(conf-macl-ext)# permit host 0014.2211.1112 any vlan 101 arp arp-guard
device(conf-macl-ext)# deny any any arp
device(conf-macl-ext)# permit any any
```

4. Enter the **interface** command.

```
device(conf)# interface ethernet 1/2
```

5. Enter the **switchport** command.

```
device(conf-if-eth-1/2)# switchport
```

6. Enter the **mac access-group** command, specifying the ACL and the **in** direction.

```
device(conf-if-eth-1/2)# mac access-group arp_guard_enable_1 in
```

Implementing rate limiting for ARP Guard

Use this procedure to implement ACL-based rate limiting for ARP Guard on a physical interface or a port-channel.

NOTE
For details on ACL-based rate limiting, refer to the *Extreme SLX-OS QoS and Traffic Management Configuration Guide*.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **mac access-list extended** command to create an ACL.

```
device(config)# mac access-list extended rate-limit-acl_1
```

3. Create needed ACL rules.

```
device(conf-macl-ext)# permit host 0014.2211.1111 any vlan 100 arp arp-guard
device(conf-macl-ext)# permit host 0014.2211.1112 any vlan 101 arp arp-guard
device(conf-macl-ext)# deny any any arp
device(conf-macl-ext)# permit any any
```

4. Create a class map.

```
device(config)# class-map arp-guard-class
```

5. While in class map mode, associate the class map with the ACL.

```
device(config-classmap)# match access-group rate-limit-acl_1
```

6. Return to global configuration mode and then create a policy map.

```
device(config-classmap)# exit
device(config)# policy-map arp-guard-pmap
```

7. Associate the class map with the policy map.

```
device(config-policymap)# class arp-guard-class
```

8. Specify class map policer parameters.

```
device(config-policymap-class)# police cir 100000
```

9. Return to global configuration mode and then enter configuration mode for the relevant interface.

```
device(config-policymap-class-police)# end
device# configure terminal
device(config)# interface ethernet 1/2
```

10. Bind the policy map to the interface.

```
device(conf-if-eth-1/2)# service-policy in arp-guard-pmap
```

ARP Guard show commands

There is one ARP show command that displays information relevant to ARP Guard.

TABLE 10 ARP Guard show commands in the *SLX-OS Command Reference*

Command	Description
show arp access-list	For one or all ARP ACLs defined on a device, displays ACL names and their permit statements.

ARP and ND suppression

In a data center fabric, the ARP and ND suppression features can help reduce ARP and ND control traffic.

> NOTE
> This feature is supported on VLANs.

> NOTE
> This feature is supported on VLANs and bridge domains (BDs).

The default scenario (disablement of ARP and ND suppression) can lead to excess control traffic:

1. A device needs to forward traffic to an IP address, but the corresponding MAC address is not in its ARP or ND cache.
2. The device broadcasts an ARP or ND request throughout the IP Fabric.

When ARP and ND suppression are enabled, excess control traffic is reduced:

1. A device needs to forward traffic to an IP address, but the corresponding MAC address is not in its ARP or ND cache.
2. The device broadcasts an ARP or ND request.
3. The leaf BGP EVPN control plane intercepts the request and looks for a match in its local cache.
   - If there is a match, the control plane responds to the device (rather than broadcasting the original request to the entire IP Fabric).
   - Only if there is no match, does the leaf control plane broadcast the request to the entire IP Fabric.

> NOTE
> When the static anycast gateway feature is enabled in an IP Fabric, to prevent ARP/ND broadcast across the network, the user should enable ARP/ND suppression on the respective VLANs.

NOTE
When the static anycast gateway feature is enabled in an IP Fabric, to prevent ARP/ND broadcast across the network, the user should enable ARP/ND suppression—disabled by default—on the respective VLANs or BDs.

Supported platforms

The ARP and ND suppression features are supported on the following platforms:

- ExtremeSwitching VDX 6740 series
- ExtremeSwitching VDX 6940 series
- ExtremeSwitching VDX 2741 (not supported for IP Fabrics)
- ExtremeSwitching VDX 2746 (not supported for IP Fabrics)

Enabling ARP and ND suppression on a VLAN

Use this procedure to enable and disable ARP and ND suppression on a VLAN.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface vlan** command to access VLAN configuration mode.

   ```
   device(config)# interface vlan 110
   ```

3. Enter the **vlan** command to access VLAN configuration mode.

   ```
   device(config)# vlan 110
   ```

4. To enable ARP suppression, enter **suppress-arp**.

   ```
   device(config-Vlan-110)# suppress-arp
   ```

   To disable ARP suppression, enter **no suppress-arp**.

5. To enable ND suppression, enter **suppress-nd**.

   ```
   device(config-Vlan-110)# suppress-nd
   ```

   To disable ND suppression, enter **no suppress-nd**.

The following example enables ARP suppression on VLAN 110.

```
device# configure terminal
device(config)# interface vlan 110
device(config-Vlan-110)# suppress-arp

device# configure terminal
device(config)# vlan 110
device(config-Vlan-110)# suppress-arp
```

Enabling ARP and ND suppression on a bridge domain

Use this procedure to enable and disable ARP and ND suppression on a bridge domain.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **bridge-domain** command to access bridge-domain configuration mode.

```
device(config)# bridge-domain 2
```

3. To enable ARP suppression, enter **suppress-arp**.

```
device(config-bridge-domain-2)# suppress-arp
```

   To disable ARP suppression, enter **no suppress-arp**.

4. To enable ND suppression, enter **suppress-nd**.

```
device(config-bridge-domain-2)# suppress-nd
```

   To disable ND suppression, enter **no suppress-nd**.

The following example enables ARP suppression on bridge domain 2.

```
device# configure terminal
device(config)# bridge-domain 2
device(config-bridge-domain-2)# suppress-arp
```

The following example enables ND suppression on bridge domain 2.

```
device# configure terminal
device(config)# bridge-domain 2
device(config-bridge-domain-2)# suppress-nd
```

ARP and ND suppression show and clear commands

There is a full range of ARP and ND suppression **show** and **clear** commands, listed here with descriptions.

TABLE 11 ARP and ND suppression show commands in the *Command Reference*

Command	Description
show ip arp suppression-cache	Displays IPv4 ARP suppression information.
show ip arp suppression-statistics	Displays IPv4 ARP suppression statistics.
show ip arp suppression-status	Displays the IPv4 ARP suppression status.
show ipv6 nd suppression-cache	Displays IPv6 ND suppression information.
show ipv6 nd suppression-statistics	Displays IPv6 ND suppression statistics.
show ipv6 nd suppression-status	Displays the IPv6 ND suppression status.

TABLE 12 ARP and ND suppression clear commands in the *Command Reference*

Command	Description
clear ip arp suppression-cache	Clears the IPv4 ARP suppression cache. You can also clear the cache for a specified VLAN.
clear ip arp suppression-statistics	Clears the IPv4 ARP suppression statistical information. You can also clear statistics for a specified VLAN.
clear ipv6 nd suppression-cache	Clears the IPv6 ND suppression cache. You can also clear the cache for a specified VLAN.
clear ipv6 nd suppression-statistics	Clears the IPv6 ND suppression statistical information. You can also clear statistics for a specified VLAN.

TABLE 13 ARP and ND suppression clear commands in the *Command Reference*

Command	Description
clear ip arp suppression-cache	Clears the IPv4 ARP suppression cache. You can also clear the cache for a specified VLAN or bridge domain.
clear ip arp suppression-statistics	Clears the IPv4 ARP suppression statistical information. You can also clear statistics for a specified VLAN or bridge domain.
clear ipv6 nd suppression-cache	Clears the IPv6 ND suppression cache. You can also clear the cache for a specified VLAN or bridge domain.
clear ipv6 nd suppression-statistics	Clears the IPv6 ND suppression statistical information. You can also clear statistics for a specified VLAN or bridge domain.

Conversational ARP and ND

Conversational ARP and ND reduce the number of cached ARP and ND entries by programming only active flows into the forwarding plane. This feature helps to optimize utilization of hardware resources.

In many use-case scenarios—especially in an IP Fabric—there are software requirements for ARP and ND entries beyond the hardware capacity. Conversational ARP and ND limit storage-in-hardware to active ARP and ND entries; aged-out entries are deleted automatically.

> **NOTE**
> For the current release, Conversational ARP and ND are not supported in an IP Fabric.

By default, the aging-out threshold is 300 seconds. (You can change the threshold to any integer value from 60 through 100,000 seconds, either before or during enablement.) Any entry that does not have at least one conversation before aging-out is deleted from the cache. Each conversation restarts the clock for that entry.

However, aging-out is also influenced by the enablement and disablement cycle:

1. When you enable conversational ARP and ND, a fast-aging policy of 30 seconds (not configurable) applies to all entries in the ARP and ND caches at that time.
2. Under conversational ARP and ND—enabled by default—a fast-aging policy of 30 seconds (not configurable) applies to all entries in the ARP and ND caches.
3. Under conversational ARP and ND—disabled by default—a fast-aging policy of 60 seconds (not configurable) applies to all entries in the ARP and ND caches.
4. From enablement of conversational ARP and ND, the aging-time value applies for existing entries with new traffic and for new entries.
5. Upon disablement, the conversational ARP and ND timers no longer apply. All current entries become permanent as do all new entries.

The following entries are not subject to conversational behavior:

- Static ARPs and NDs
- Next hops

Static ARPs and NDs are not subject to conversational behavior.

Supported platforms

Conversational ARP and ND are supported on the following platforms:

- ExtremeSwitching VDX 6740 series
- ExtremeSwitching VDX 6940 series
- ExtremeSwitching VDX 2741 (not supported for IP Fabrics)
- ExtremeSwitching VDX 2746 (not supported for IP Fabrics)

Enabling and disabling conversational ARP and ND

Conversational ARP and ND can reduce the number of cached ARP and ND entries, optimizing utilization of hardware resources.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command to access RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 1
   ```

3. To specify an aging-time value other than the default 300 seconds, enter the **host-table aging-time conversational** command.

   ```
   device(config-rbridge-id-1)# host-table aging-time conversational 600
   device(config)# host-table aging-time conversational 600
   ```

   To restore the default aging-time value of 300 seconds, enter the **no host-table aging-time conversational** command.

4. To enable conversational ARP and ND, enter the **host-table aging-mode conversational** command.

   ```
   device(config-rbridge-id-1)# host-table aging-mode conversational
   device(config)# host-table aging-mode conversational
   ```

   To disable conversational ARP and ND, enter the **no host-table aging-mode conversational** command.

The following example implements conversational ARP and ND. The current aging-time value applies.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# host-table aging-mode conversational

device# configure terminal
device(config)# host-table aging-mode conversational
```

Conversational ARP and ND show and clear commands

Conversational ARP and ND **show** and **clear** commands are listed here with descriptions.

TABLE 14 Conversational ARP and ND show commands in the *Command Reference*

Command	Description
show arp	Displays the ARP cache.
show ipv6 neighbor	Displays IPv6 neighbor information.

TABLE 15 Conversational ARP and ND clear commands in the *Command Reference*

Command	Description
clear arp	Clears the local ARP cache and then resends ARP requests to the local hosts.
clear arp no-refresh	Clears the ARP cache, without resending ARP requests to the local hosts.
clear ipv6 neighbor	Clears the IPv6 ND cache and then resends ND requests to the local hosts.
clear ipv6 neighbor no-refresh	Clears the ND cache, without resending ND requests to the local hosts.

IP Addressing

- IP addressing overview....49
- Basic IP parameters and defaults....51
- Assigning IP addresses to interfaces....59
- IP directed broadcast on an interface....61
- Router-ID IP addresses....61
- DNS....62
- Source IP address for various packet types....64
- IPv4 MTU settings....65
- Configuring the maximum ICMP error message rate....66
- IP addressing Show and Clear commands....69

IP addressing overview

IPv4 uses a 32-bit addressing system designed for use in packet-switched networks. IPv4 routing is enabled by default on Network OS devices that operate at Layer 3 and cannot be disabled.

IPv4 uses a 32-bit addressing system designed for use in packet-switched networks. IPv4 routing is enabled by default on SLX-OS devices that operate at Layer 3 and cannot be disabled.

IPv4 is an Internet protocol used to deliver packets of data from a source to a destination across an interconnected system of networks. IPv4 uses a fixed-length 32-bit addressing system and is represented in a 4-byte dotted decimal format: x.x.x.x.

IP uses four main mechanisms to provide service:

- Type of Service (ToS)—Indicates the Quality of Service (QoS) required for a specific traffic type or network and enables a higher priority to be given to voice traffic, for example, that is more sensitive to dropped packets.
- Time to Live (TTL)—The time period for which a packet can exist before it reaches its final destination. If the TTL expires before the packet reaches its destination, the packet is destroyed. The period is set by the packet sender.
- Options—Control mechanisms such as timestamps, security, and other special routing functions that are optional.
- Header Checksum—Used to verify that the packet contents have transmitted correctly. If the checksum algorithm fails, the packet is dropped immediately.

IP does not provide a reliable communication function. No acknowledgments are sent and the only error control is the header checksum. There are not flow-control mechanisms or retransmissions. Errors detected may be reported via the Internet Control Message Protocol (ICMP).

IPv4 address classes

An IP address is divided into two sections:

- Network—Identifies the network on which the device is configured.
- Host—Identifies the host device.

There are three main classes of IP address as shown in the table with the number of bits that are allocated for the network and host sections of the IPv4 address. Both classical IP network masks and Classless Interdomain Routing (CIDR) network masks are supported.

Class	Network bits	Host bits	Example network address with classical mask	Example network address with CIDR mask
A	8	24	10.0.0.0 255.0.0.0	10.0.0.0/24
B	16	16	172.16.0.0 255.255.0.0	172.16.0.0/16
C	24	8	192.168.254.0 255.255.255.0	192.168.254.0/8

IPv4 address classes

An IP address is divided into two sections:

- Network—Identifies the network on which the device is configured.
- Host—Identifies the host device.

IP Interfaces

NetIron devices operating at Layer 3 allow IP addresses to be configured on the following types of interfaces:

- Ethernet ports
- Virtual routing interfaces (used by VLANs)
- Loopback interfaces
- GRE tunnels

You can configure up to 24 IP addresses on each interface. You can increase the amount of IP subnet addresses to 128 by increasing the size of the ip-subnet-port table.

> **NOTE**
> After you configure a virtual routing interface on a VLAN, you cannot configure Layer 3 interface parameters on individual ports. The parameters must be configured on the appropriate virtual routing interface.

IP Interfaces

SLX-OS devices operating at Layer 3 allow IP addresses to be configured on the following types of interfaces:

- Ethernet ports
- Virtual routing interfaces
- Loopback interfaces

You can configure up to 128 IP addresses on each interface.

> **NOTE**
> After you configure a port as a "switchport," you cannot configure Layer 3 interface parameters on that port. The parameters must be configured on the appropriate virtual routing interface.

The ARP cache

The ARP cache contains entries that map IP addresses to MAC addresses.

ARP cache entries are added in one of the following ways:

- From devices that are directly attached to the Layer 3 device.

- From an interface-based static IP route that goes to a destination two or more router hops away. The MAC address is either of the destination device of the router interface answering an ARP request on behalf of the device, using proxy ARP.

The ARP cache can contain both dynamic (learned) entries and static (user-configured) entries. The software places an entry in the ARP cache:

- Dynamic—when the Layer 3 device learns a device MAC address from an ARP request or ARP reply from the device.
- Static—When the interface with proper IP address comes up.

31-bit subnet masks on point-to-point networks

To conserve IPv4 address space, a 31-bit subnet mask can be assigned to point-to-point networks. Support for an IPv4 address with a 31-bit subnet mask is described in RFC 3021.

With IPv4, four IP addresses with a 30-bit subnet mask are allocated on point-to-point networks. In contrast, a 31-bit subnet mask uses only two IP addresses: all zero bits and all one bits in the host portion of the IP address. The two IP addresses are interpreted as host addresses, and do not require broadcast support because any packet that is transmitted by one host is always received by the other host at the receiving end. Therefore, directed broadcast on a point-to-point interface is eliminated.

IP-directed broadcast CLI configuration at the global level, or the per interface level, is not applicable on interfaces configured with a 31-bit subnet mask IP address.

When the 31-bit subnet mask address is configured on a point-to-point link, using network addresses for broadcast purposes is not allowed. For example, in an IPV4 broadcast scheme, the following subnets can be configured:

- 10.10.10.1 - Subnet for directed broadcast: {*Network-number*, -1}
- 10.10.10.0 - Subnet for network address: {*Network-number*, 0}

In a point-to-point link with a 31-bit subnet mask, the previous two addresses are interpreted as host addresses and packets are not rebroadcast.

IP unnumbered interfaces

With large numbers of routers and redundant Layer 3 interfaces, a significant number of IP addresses are consumed just to configure the network itself. Using /31 masks reduces the consumption of addresses, but two IP addresses are still consumed per interface.

This feature borrows an IP address from another Layer 3 interface already configured on the router. This is particularly useful in a typical Layer 3 Clos network, where routers are directly connected and require the assignment of IP addresses to each pair of routers, typically from the same subnet.

For further information, see IP unnumbered interface.

Basic IP parameters and defaults

The following protocols are disabled by default:

- Route exchange protocols (RIP, OSPF, IS-IS, BGP4)
- Route exchange protocols (OSPF, IS-IS, BGP4)
- Route exchange protocols (OSPF, BGP4)
- Route exchange protocols (OSPF, BGP4)
- Multicast protocols (IGMP, PIM-DM, PIM-SM)
- Router redundancy protocols (VRRP-E, VRRP, FSRP)

- Multicast protocols (IGMP, PIM-SM)
- Multicast protocols
- Router redundancy protocols (VRRP-E, VRRP)

Parameter changes in effect

Most IP parameters described in this chapter are dynamic. They take effect immediately, as soon as you enter the CLI command. You can verify that a dynamic change has taken effect by displaying the running configuration.

To display the running configuration, enter the **show running-config** command.

To save a configuration change permanently so that the change remains in effect following a system reset or software reload, save the configuration to the startup configuration file.

To change the memory allocation, you must reload the software after you save the changes to the startup configuration file.

IP global parameters

The following table lists the IP global parameters, their default values, and where to find configuration information.

TABLE 16 IP global parameters

Parameter	Description	Default
IP state	The Internet Protocol, version 4	Enabled **Note:** You cannot disable IP.
IP address and mask notation	Format for displaying an IP address and its network mask information. You can enable one of the following: • Class-based format; example: 192.168.1.1 255.255.255.0 • Classless Interdomain Routing (CIDR) format; example: 192.168.1.1/24	Class-based **Note:** Changing this parameter affects the display of IP addresses, but you can enter addresses in either format regardless of the display setting.
Router ID	The value that routers use to identify themselves to other routers when exchanging route information. OSPF and BGP4 use router IDs to identify routers. RIP does not use the router ID.	The IP address configured on the lowest-numbered loopback interface. If no loopback interface is configured, then the lowest-numbered IP address configured on the device.
IP Maximum Transmission Unit (MTU)	The maximum length an Ethernet packet can be without being fragmented.	1500 bytes for Ethernet II encapsulation 1492 bytes for SNAP encapsulation
Address Resolution Protocol (ARP)	A standard IP mechanism that routers use to learn the Media Access Control (MAC) address of a device on the network. The router sends the IP address of a device in the ARP request and receives the device's MAC address in an ARP reply.	Enabled
ARP rate limiting	Lets you specify a maximum number of ARP packets the device will accept each second. If the device receives more ARP packets than you specify, the device drops additional ARP packets for the remainder of the one-second interval.	Disabled
ARP age	The amount of time the device keeps a MAC address learned through ARP in the device's ARP cache. The device resets the timer to zero	Ten minutes

TABLE 16 IP global parameters (continued)

Parameter	Description	Default
	each time the ARP entry is refreshed and removes the entry if the timer reaches the ARP age. **Note:** You also can change the ARP age on an individual interface basis. Refer to IP interface parameters on page 57.	
Proxy ARP	An IP mechanism a router can use to answer an ARP request on behalf of a host, by replying with the interface's own MAC address instead of the host's.	Disabled
Static ARP entries	An ARP entry you place in the static ARP table. Static entries do not age out.	No entries
Time to Live (TTL)	The maximum number of routers (hops) through which a packet can pass before being discarded. Each router decreases a packet's TTL by 1 before forwarding the packet. If decreasing the TTL causes the TTL to be 0, the router drops the packet instead of forwarding it.	64 hops
Directed broadcast forwarding	A directed broadcast is a packet containing all ones (or in some cases, all zeros) in the host portion of the destination IP address. When a router forwards such a broadcast, it sends a copy of the packet out each of its enabled IP interfaces. **Note:** You also can enable or disable this parameter on an individual interface basis. Refer to IP interface parameters on page 57.	Disabled
Directed broadcast mode	The packet format the router treats as a directed broadcast. The following formats can be directed broadcast: • All ones in the host portion of the packet's destination address. • All zeroes in the host portion of the packet's destination address.	All ones **Note:** If you enable all-zeroes directed broadcasts, all-ones directed broadcasts remain enabled.
Source-routed packet forwarding	A source-routed packet contains a list of IP addresses through which the packet must pass to reach its destination.	Enabled
Internet Control Message Protocol (ICMP) messages	The Extreme device can send the following types of ICMP messages: • Echo messages (ping messages) • Destination Unreachable messages • Redirect messages **Note:** You also can enable or disable ICMP Redirect messages on an individual interface basis. Refer to IP interface parameters on page 57.	Enabled
ICMP Router Discovery Protocol (IRDP)	An IP protocol a router can use to advertise the IP addresses of its router interfaces to directly attached hosts. You can enable or disable the	Disabled

TABLE 16 IP global parameters (continued)

Parameter	Description	Default
	protocol, and change the following protocol parameters: • Forwarding method (broadcast or multicast) • Hold time • Maximum advertisement interval • Minimum advertisement interval • Router preference level **Note:** You also can enable or disable IRDP and configure the parameters on an individual interface basis. Refer to IP interface parameters on page 57.	
Maximum BootP relay hops	The maximum number of hops away a BootP server can be located from a router and still be used by the router's clients for network booting.	Four
Domain name for Domain Name Server (DNS) resolver	A domain name (example: extreme.router.com) you can use in place of an IP address for certain operations such as IP pings, trace routes, and Telnet management connections to the device.	None configured
DNS default gateway addresses	A list of gateways attached to the device through which clients attached to the device can reach DNS.	None configured
IP load sharing	A feature that enables the device to balance traffic to a specific destination across multiple equal-cost paths. Load sharing is based on a combination of destination MAC address, source MAC address, destination IP address, source IP address, and IP protocol. **Note:** Load sharing is sometimes called Equal Cost Multi Path (ECMP).	Enabled
Maximum IP load sharing paths	The maximum number of equal-cost paths across which the Extreme device is allowed to distribute traffic.	Four
Origination of default routes	You can enable a device to originate default routes for the following route exchange protocols, on an individual protocol basis: • RIP • OSPF • BGP4	Disabled
Default network route	The device uses the default network route if the IP route table does not contain a route to the destination and also does not contain an explicit default route (0.0.0.0 0.0.0.0 or 0.0.0.0/0).	None configured
Static route	An IP route you place in the IP route table.	No entries
Source interface	The IP address the device uses as the source address for Telnet, RADIUS, or TACACS/TACACS+ packets originated by the device. The	The lowest-numbered IP address on the interface the packet is sent on.

TABLE 16 IP global parameters (continued)

Parameter	Description	Default
	device can select the source address based on either of the following: • The lowest-numbered IP address on the interface the packet is sent on. • The lowest-numbered IP address on a specific interface. The address is used as the source for all packets of the specified type regardless of interface the packet is sent on.	
Parameter	Description	Default
IP state	The Internet Protocol, version 4	Enabled **Note:** You cannot disable IP.
Router ID	The value that routers use to identify themselves to other routers when exchanging route information. OSPF and BGP4 use router IDs to identify routers.	The IP address configured on the lowest-numbered loopback interface. If no loopback interface is configured, then the lowest-numbered IP address configured on the device.
IP Maximum Transmission Unit (MTU)	The maximum length an Ethernet packet can be without being fragmented.	1500 bytes for Ethernet II encapsulation
Address Resolution Protocol (ARP)	A standard IP mechanism that routers use to learn the Media Access Control (MAC) address of a device on the network. The router sends the IP address of a device in the ARP request and receives the device's MAC address in an ARP reply.	Enabled
ARP rate limiting	Lets you specify a maximum number of ARP packets the device will accept each second. If the device receives more ARP packets than you specify, the device drops additional ARP packets for the remainder of the one-second interval.	Disabled
ARP age	The amount of time the device keeps a MAC address learned through ARP in the device's ARP cache. The device resets the timer to zero each time the ARP entry is refreshed and removes the entry if the timer reaches the ARP age. **Note:** You also can change the ARP age on an individual interface basis. Refer to IP interface parameters on page 57.	25 minutes
Proxy ARP	An IP mechanism a router can use to answer an ARP request on behalf of a host, by replying with the router's own MAC address instead of the host's.	Enabled
Static ARP entries	An ARP entry you place in the static ARP table. Static entries do not age out.	No entries
Time to Live (TTL)	The maximum number of routers (hops) through which a packet can pass before being discarded. Each router decreases a packet's TTL by 1 before forwarding the packet. If decreasing the TTL causes the TTL to be 0, the router drops the packet instead of forwarding it.	64 hops

Parameter	Description	Default
Directed broadcast forwarding	A directed broadcast is a packet containing all ones (or in some cases, all zeros) in the host portion of the destination IP address. When a router forwards such a broadcast, it sends a copy of the packet out each of its enabled IP interfaces. **Note:** You also can enable or disable this parameter on an individual interface basis. Refer to IP interface parameters on page 57.	Disabled
Source-routed packet forwarding	A source-routed packet contains a list of IP addresses through which the packet must pass to reach its destination.	Enabled
Internet Control Message Protocol (ICMP) messages	The Extreme device can send the following types of ICMP messages: • Echo messages (ping messages) • Destination Unreachable messages • Redirect messages **Note:** You also can enable or disable ICMP Redirect messages on an individual interface basis. Refer to IP interface parameters on page 57.	Enabled
Maximum DHCP relay hops	The maximum number of hops away a BootP server can be located from a router and still be used by the router's clients for network booting.	Four
Domain name for Domain Name Server (DNS) resolver	A domain name (example: extreme.router.com) you can use in place of an IP address for certain operations such as IP pings, trace routes, and Telnet management connections to the device.	None configured
DNS default gateway addresses	A list of gateways attached to the device through which clients attached to the device can reach DNS.	None configured
IP load sharing	A feature that enables the device to balance traffic to a specific destination across multiple equal-cost paths. Load sharing is based on a combination of destination MAC address, source MAC address, destination IP address, source IP address, and IP protocol. **Note:** Load sharing is sometimes called Equal Cost Multi Path (ECMP).	Enabled
Maximum IP load sharing paths	The maximum number of equal-cost paths across which this device is allowed to distribute traffic.	64
Origination of default routes	You can enable a device to originate default routes for the following route exchange protocols, on an individual protocol basis: • OSPF • BGP4 • IS-IS	Disabled
Static route	An IP route you place in the IP route table.	No entries
Source interface	The IP address the device uses as the source address for Telnet, RADIUS, or TACACS/	The lowest-numbered IP address on the interface the packet is sent on.

Parameter	Description	Default
	TACACS+ packets originated by the device. The device can select the source address based on either of the following: • The lowest-numbered IP address on the interface the packet is sent on. • The lowest-numbered IP address on a specific interface. The address is used as the source for all packets of the specified type regardless of interface the packet is sent on.	

IP interface parameters

The following table lists the interface-level IP parameters, their default values, and where to find configuration information.

TABLE 17 IP interface parameters

Parameter	Description	Default
IP state	The Internet Protocol, version 4	Enabled **Note:** You cannot disable IP.
IP address	A Layer 3 network interface address The device has separate IP addresses on individual interfaces.	None configured
Encapsulation type	The format of the packets in which the device encapsulates IP datagrams. The encapsulation format can be one of the following: • Ethernet II • SNAP	Ethernet II
IP Maximum Transmission Unit (MTU)	The maximum length (number of bytes) of an encapsulated IP datagram the device can forward.	1500 for Ethernet II encapsulated packets 1492 for SNAP encapsulated packets
ARP age	Locally overrides the global setting. Refer to IP global parameters on page 52.	Ten minutes
Directed broadcast forwarding	Locally overrides the global setting. Refer to IP global parameters on page 52.	Disabled
ICMP Router Discovery Protocol (IRDP)	Locally overrides the global IRDP settings. Refer to IP global parameters on page 52.	Disabled
ICMP Redirect messages	Locally overrides the global setting. Refer to IP global parameters on page 52.	Enabled
DHCP gateway stamp	The device can assist DHCP/BootP Discovery packets from one subnet to reach DHCP/BootP servers on a different subnet by placing the IP address of the device interface that receives the request in the request packet's Gateway field. You can override the default and specify the IP address to use for the Gateway field in the packets. **Note:** UDP broadcast forwarding for client DHCP/BootP requests (bootpc) must be enabled and you must configure an IP helper address (the server's IP address or a directed	The lowest-numbered IP address on the interface that receives the request

TABLE 17 IP interface parameters (continued)

Parameter	Description	Default
	broadcast to the server's subnet) on the port connected to the client.	
UDP broadcast forwarding	The device can forward UDP broadcast packets for UDP applications such as BootP. By forwarding the UDP broadcasts, the device enables clients on one subnet to find servers attached to other subnets. **Note:** To completely enable a client's UDP application request to find a server on another subnet, you must configure an IP helper address consisting of the server's IP address or the directed broadcast address for the subnet that contains the server. Refer to the next row.	The device helps forward broadcasts for the following UDP application protocols: • bootps • dns • netbios-dgm • netbios-ns • tacacs • tftp • time
IP helper address	The IP address of a UDP application server (such as a BootP or DHCP server) or a directed broadcast address. IP helper addresses allow the device to forward requests for certain UDP applications from a client on one subnet to a server on another subnet.	None configured
Parameter	**Description**	**Default**
IP state	The Internet Protocol, version 4	Enabled **Note:** You cannot disable IP.
IP address	A Layer 3 network interface address The SLX-OS device has separate IP addresses on individual interfaces.	None configured
Encapsulation type	The format of the packets in which the device encapsulates IP datagrams. The encapsulation format can be one of the following: • Ethernet	Ethernet
IP Maximum Transmission Unit (MTU)	The maximum length (number of bytes) of an encapsulated IP datagram the device can forward.	1500 for Ethernet II encapsulated packets
ARP age	Locally overrides the global setting. Refer to IP global parameters on page 52.	25 minutes
Directed broadcast forwarding	Locally overrides the global setting. Refer to IP global parameters on page 52.	Disabled
ICMP Redirect messages	Locally overrides the global setting. Refer to IP global parameters on page 52.	Enabled
UDP broadcast forwarding	The device can forward UDP broadcast packets for UDP applications such as BootP. By forwarding the UDP broadcasts, the device enables clients on one subnet to find servers attached to other subnets. **Note:** To completely enable a client's UDP application request to find a server on another subnet, you must configure an IP helper address consisting of the server's IP address or the directed broadcast address for the subnet that contains the server. Refer to the next row.	The device helps forward broadcasts for the following UDP application protocols: • bootps • dns • netbios-dgm • netbios-ns • tacacs • tftp • time
IP helper address	The IP address of a UDP application server (such as a BootP or DHCP server) or a directed	None configured

Parameter	Description	Default
	broadcast address. IP helper addresses allow the device to forward requests for certain UDP applications from a client on one subnet to a server on another subnet.	

Assigning IP addresses to interfaces

Use the topics in this section to assign IP addresses to interfaces and to remove them.

Assigning an IP address to a loopback

IP addresses can be assigned to a loopback interface, using Classless Interdomain Routing (CIDR) network masks. Loopback interfaces add stability to a network, because they do not incur route flap problems due to unstable links between devices.

IPv4 routing is enabled by default on SLX-OS devices that operate at Layer 3 and cannot be disabled. IP addresses must be assigned to interfaces on the device to allow IPv4-based protocols to operate across the network.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Access the interface to which you are assigning the IP addresses.

```
device(config)# interface loopback 1
```

3. Assign an IP address on the interface.

   NOTE
   You can define only one IP address per loopback. The only valid mask value is **/32**.

```
device(config-Loopback-1)# ip address 1.1.1.1/32
```

4. Use the **no shutdown** command to activate the interface.

```
device(config-Loopback-1)# no shutdown
```

5. To verify that the IP address is assigned to the interface, enter the **show ip interface** command.

```
device(config-Loopback-1)# do show ip interface loopback 1
 Loopback 1 is up protocol is up
 Primary Internet Address is 1.1.1.1/32
 IP MTU is 1500
 Proxy Arp is not Enabled
 ICMP unreachables are always sent
 ICMP mask replies are never sent
 IP fast switching is enabled
 Vrf : default-vrf
```

The following example configures an IP address on a loopback interface.

```
device# configure terminal
device(config)# interface loopback 1
device(config-Loopback-1)# ip address 1.1.1.1/32
device(config-Loopback-1)# no shutdown
device(config-Loopback-1)# do show ip interface loopback 1
```

Assigning IP addresses to Ethernet interfaces

IP addresses can be assigned to an Ethernet interface, using Classless Interdomain Routing (CIDR) network masks.

Before assigning IP addresses, you must have the physical devices connected to match your IP addressing design.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Access the interface to which you are assigning the IP addresses.

```
device(config)# interface ethernet 0/1
```

3. Enter one or more IP addresses, with a CIDR network mask.

```
device(config-if-eth-0/1)# ip address 11.1.1.11/24
device(config-if-eth-0/1)# ip address 11.11.1.11/24
```

4. To assign a secondary IP address, include the **secondary** keyword.

```
device(config-if-eth-0/1)# ip address 10.53.5.4/24 secondary
```

   NOTE
   To configure a secondary IP address, you must configure the primary IP address configured within the same subnet.

5. Use the **no shutdown** command to activate the interface.

```
device(config-if-eth-0/1)# no shutdown
```

6. To verify that the IP address is assigned to the interface, enter the **show ip interface** command.

```
device(config-if-eth-0/1)# do show ip interface ethernet 3/14
 Ethernet 3/14 is up protocol is up
 Primary Internet Address is 11.1.1.11/24 broadcast is 11.1.1.255
 Primary Internet Address is 11.11.1.11/24 broadcast is 11.11.1.255
 Secondary Internet Address is 11.11.1.12/24 broadcast is 11.11.1.255
 IP MTU is 1500
 Proxy Arp is Enabled
 ICMP unreachables are always sent
 ICMP mask replies are never sent
 IP fast switching is enabled
 Vrf : default-vrf
```

Deleting an IP address from an interface

You can delete a specified IP address, or all IP addresses, from an interface.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Access the interface from which you are deleting the IP address.

```
device(config)# interface ethernet 1/5
```

3. To delete a specified IP address from the interface, enter the **no ip address** command, specifying the address and the CIDR mask.

```
device(config-if-eth-1/5)# no ip address 10.53.5.3/24
```

4. To delete all IP addresses defined on the interface, enter **no ip address**.

```
device(config-if-eth-1/5)# no ip address
```

IP directed broadcast on an interface

A directed broadcast is an IP broadcast to all devices in a directly attached network or subnet.

You can enable IP directed broadcast on a Layer 3 physical Ethernet interface or a VE interface. When the device receives a packet that has a destination IP address as Broadcast IP and IP directed broadcast is enabled on an interface through which the destination network is reachable, the interface floods the packet to all hosts of this network.

IP directed broadcast is supported on the default VRF and the user-defined VRF.

Enabling IP directed broadcast on an interface

By default, IP directed broadcast is disabled on the interfaces of the device.

1. From privileged EXEC mode, access global configuration mode.

```
device# configure terminal
```

2. Specify the interface.

```
device(config)# interface ethernet 0/2
```

3. Enable IP directed broadcast on the interface.

```
device(config-if-eth-0/2)# ip directed-broadcast
```

Router-ID IP addresses

Certain routing protocols—for example, OSPF and BGP4—have algorithms that specify which configured IP address identifies the device.

In general, a device has IP addresses assigned to various interfaces. However, some routing protocols identify the device by the router ID, rather than by the IP addresses assigned to the interfaces connected by the protocol. If a router ID is not specified, such protocols use the following algorithm to select a router ID:

1. If the device has loopback interfaces, the router ID is the IP address configured on the lowest numbered loopback interface. For example, if you configure loopback interfaces 1, 2, and 3 as follows, the default router ID is 10.9.9.9/32:
   - Loopback 1: 10.9.9.9/32
   - Loopback 2: 10.4.4.4/32
   - Loopback 3: 10.1.1.1/32
2. If a loopback interface is not configured, then the lowest IP address configured on a physical interface becomes the router ID.

You can also specify a router-id IP address, using the **ip router-id** command.

Assigning a router-ID IP address

For routing protocols that decide which configured IP address identifies the device—for example, OSPF and BGP4—an IP address that you apply under global configuration takes precedence.

NOTE
If you change the router ID, all current BGP4 sessions are cleared.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ip router-id** command, specifying the device-level IP address.

```
device(config)# ip router-id 10.11.12.13
```

NOTE
You can specify an IP address that is also assigned to an interface on that device. But make sure that the router-ID IP address is not assigned on another network device.

DNS

The Domain Name System (DNS) operates as a hierarchical naming system that assigns a name (such as a company name) to an Internet entity to represent the real IP address of the entity. An entity can be a gateway router and is referred to as a domain.

A domain name (for example, extreme.router.com) can be used in place of an IP address for certain operations such as IP pings, trace routes, and Telnet management connections to the router. Instead of having to remember all the numbers of the IP address, a domain name is easier to remember. DNS is comprised of the following:

- DNS Server
- DNS Resolver
- DNS gateway addresses

DNS Server

A DNS server stores the information about a DNS domain. DNS servers are a key element of DNS because they respond to queries against its database. When a DNS domain is defined on this device to recognize all hosts within that domain, this device automatically appends the appropriate domain to the host address and forwards it to the domain name server.

DNS Resolver

The DNS resolver is a feature in a Layer 2 or Layer 3 device that sends and receives queries to and from the DNS server on behalf of a client. You can create a list of domain names that can be used to resolve host names. This list can have more than one domain name. When a client performs a DNS query, all hosts within the domains in the list can be recognized and queries can be sent to any domain on the list. After you define a domain name, the device automatically appends the appropriate domain to a host and forwards it to the DNS servers for resolution.

DNS Gateway Addresses

Gateway IP addresses assigned to the device through enable clients attached to the device to reach DNS.

Configuring DNS

A Domain Name System (DNS) domain and DNS gateway addresses can be configured to resolve host names to IP addresses.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Use the **ip dns domain-name** to configure a domain name.

```
device(config)# ip dns domain-name mycompany.com

device(config)# ip dns domain-list mycompany.com

device(config)# (config)# ip dns domain-name mycompany.com
Warning : Name Server also needs to be configured for successful operation
```

3. Configure up to four default DNS gateway addresses for DNS servers.

```
device(config)# ip dns server-address 10.157.22.199 10.96.7.15 10.95.7.25 10.98.7.15
```

   The first entry serves as the primary default gateway address.

4. Use the **ip dns name-server** command to configure the default DNS gateway address for DNS servers.

```
device(config)# ip dns name-server 10.157.22.199
```

5. Use the **ip dns name-server** command to configure the secondary DNS gateway address for DNS servers.

```
device(config)# ip dns name-server 10.157.22.199
```

6. Use the **exit** command to return to privileged EXEC mode.

```
device(config)# exit
```

7. (Optional) Use the **traceroute** command to verify the DNS configuration.

```
device# traceroute mycompany.com

Sending DNS Query to 10.157.22.199
Tracing Route to IP node 10.157.22.80
To ABORT Trace Route, Please use stop-traceroute command.
 Traced route to target IP node 10.157.22.80:
  IP Address    Round Trip Time1    Round Trip Time2
  10.95.6.30    93 msec             121 msec
```

   The output shows that 10.157.22.199 is the IP address of the domain name server (default DNS gateway address), and 10.157.22.80 represents the IP address of the mycompany.com host.

The following example configures a DNS domain and four default gateways. The output from the **traceroute**command shows that 10.157.22.199 is the IP address of the domain name server (default DNS gateway address), and 10.157.22.80 represents the IP address of the mycompany.com host.

```
device# configure terminal
device(config)# ip dns domain-name mycompany.com
device(config)# ip dns server-address 10.157.22.199 10.96.7.15 10.95.7.25 10.98.7.15
device(config)# exit
device# traceroute mycompany.com

Sending DNS Query to 10.157.22.199
Tracing Route to IP node 10.157.22.80
To ABORT Trace Route, Please use stop-traceroute command.
 Traced route to target IP node 10.157.22.80:
  IP Address    Round Trip Time1    Round Trip Time2
  10.95.6.30    93 msec             121 msec
```

The following example configures a DNS domain; and default and secondary DNS gateway addresses for DNS servers

```
device# configure terminal
device(config)# ip dns domain-name www.mycompany.com
device(config)# ip dns name-server 10.157.22.199
device(config)# exit
device(config)# ip dns name-server 10.96.7.15
```

Source IP address for various packet types

When a device originates a packet of one of the following types, the default source address of the packet is the lowest-numbered IP address on the interface that sends the packet:

- Telnet
- TACACS/TACACS+
- TFTP
- RADIUS
- Syslog
- SNTP

You can configure the device to always use the lowest-numbered IP address on a specific Ethernet, loopback, or virtual interface as the source addresses for these packets. When configured, the device uses the same IP address as the source for all packets of the specified type, regardless of the ports that actually sends the packets.

Designating a source IP address provides the following benefits:

- If your server is configured to accept packets only from specific IP addresses, you can configure the device to always send the packets from the same link or source address.
- If you specify a loopback interface as the single source for specified packets, servers can receive the packets regardless of the states of individual links. Thus, if a link to the server becomes unavailable, but can be reached through another link, the client or server still receives the packets, and the packets still have the source IP address of the loopback interface.

IPv4 MTU settings

By increasing the IPv4 maximum transmission unit (MTU), you can reduce packet fragmentation. You can configure IP MTU only on L3 interfaces such as VE port or router port.

The IPv4 MTU is the maximum length of an IPv4 packet that a Layer 2 frame can contain. If an IPv4 packet is larger than the MTU allowed by the frame, the device fragments the IP packet into multiple parts that will fit into frames, and sends the parts of the fragmented IP packet separately, in different frames. The device that receives the multiple fragments of the IP packet reassembles the fragments into the original packet. The default IPv4 MTU is 1500 bytes for Ethernet II packets.

There are limitations on IP MTU. The hardware only supports 3 profiles, including the default value of 1500 and two user-defined values. The previous lower supported value is used for traffic coming through the device when you configure the MTU above these values.

There are limitations on IP MTU and even if you set specific values, the hardware only supports 3 profiles - 1300, 1500, 9194. The previous lower supported value is used for traffic coming through the device when you configure the MTU above these values.

You can change the MTU for individual IP interfaces. However, IPv4 MTU cannot be set higher than the maximum frame size, minus 18.

> **NOTE**
> For multicast data traffic, frames are not fragmented and the IP MTU setting is ignored.

For jumbo packets, the device supports hardware forwarding. Unicast jumbo packets received on a port that supports the frame's IPv4 MTU size and forwarded to another port that also supports the frame's MTU size are forwarded in hardware.

IPv4 MTU and maximum frame size

Because raising MTU demands system resources, increase MTU only on the IP interfaces that need it.

For example, if you have one interface connected to a server that uses jumbo frames and two other interfaces connected to clients that can support the jumbo frames, you might increase the MTU only on those three IP interfaces.

Because you need to allow for the Ethernet header and CRC, IP MTU must always be less that the maximum frame size, as follows:

- 18 bytes for untagged packets
- 22 bytes for single-tagged packets
- 26 bytes for dual-tagged packets

Changing IPv4 MTU on an interface

You can change the size of the IPv4 maximum transmission unit (MTU) on each L3 interface. IP MTU configuration is not supported on a port-channel.

By default, IPv4 MTU is 1500 bytes. The IPv4 MTU ranges from 1300 through 9194 bytes.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Access the interface on which you need to modify the IPv4 MTU.

```
device(config)# interface ethernet 0/1
device(config)# interface ethernet 1/5
```

3. Enter the **ip mtu** command, specifying the relevant value.

```
device(config-if-eth-0/1)# ip mtu 2000

device(config-if-eth-1/5)# ip mtu 2000
```

The following example changes the IPv4 MTU on an Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ip mtu 2000

device# configure terminal
device(config)# interface ethernet 1/5
device(config-if-eth-1/5)# ip mtu 2000
```

NOTE
If the interface is part of a VE, change the IPv4 MTU only at the VE interface and not at the physical port. Switchports have only the MTU value configured. They do not inherit from IP MTU of Ve. Extreme recommends that you configure both MTU and IP MTU to be the same value.

To change the IPv4 MTU at the VE interface, enter the following commands:

```
device# configure terminal
device(config)# interface ve 103
device(config-ve-103)# ip mtu 2000
```

Configuring the maximum ICMP error message rate

NOTE
The maximum ICMP error message rate configuration only supports IPv4 traffic.

The Extreme device configuration allows 200 ICMP error messages per second per IP interface. You can now configure the maximum ICMP error message rate on all Interface Modules. The maximum configured value is increased to 5000 error messages per second. The maximum ICMP error message rate configuration uses an ICMP error metering mechanism. The process for the ICMP error metering mechanism is as follows:

- There is a meter counter for each interface. There is one total meter counter per Interface Module.
- The interface counter and the total counter will increment every time an icmp error message is sent out.
- The timer will rest all counters to 0 every second.
- Before an error message is sent out, it check the interface meter counter against the user configured icmp error limit (5000 max). The total counter will check against 10000. The error message is dropped if one any counter is larger the checked value.

The total error rate for all IP interfaces on an Interface Module is 10,000 errors per second. The ICMP error metering mechanism is per IP interface; this includes VRF IP interfaces.

Since the ICMP error metering code implementation is similar between the Management Module and Interface Module code, this change will also affect the Management Module ICMP error rate.

To configure the maximum ICMP error rate, enter the following command.

```
device(config)# ip icmp max-err-msg-rate 600
```

Syntax: [no] ip icmp max-err-msg-rate error per second

The *error per second* variable specifies the maximum error rate in errors per second. The maximum configured value has a range from 0 (minimum) to 5000 (maximum) error message per second. The default value is 400.

Disabling ICMP messages

The Extreme device is enabled to reply to ICMP echo messages and send ICMP Destination Unreachable messages by default.

You can selectively disable the following types of Internet Control Message Protocol (ICMP) messages:

- **Echo messages (ping messages)** - The Extreme device replies to IP pings from other IP devices.
- **Destination Unreachable messages** - If the Extreme device receives an IP packet that it cannot deliver to its destination, the Extreme device discards the packet and sends a message back to the device that sent the packet. The message informs the device that the destination cannot be reached by the Extreme device.

Disabling replies to broadcast ping requests

By default, the Extreme device is enabled to respond to broadcast ICMP echo packets, which are ping requests.

To disable response to broadcast ICMP echo packets (ping requests), enter the following command.

```
device(config)# no ip icmp echo broadcast-request
```

Syntax: [no] ip icmp echo broadcast-request

If you need to re-enable response to ping requests, enter the following command.

```
device(config)# ip icmp echo broadcast-request
```

Disabling ICMP destination unreachable messages

By default, when this Extreme device receives an IP packet that the device cannot deliver, the device sends an ICMP Unreachable message back to the host that sent the packet. You can selectively disable a device's response to the following types of ICMP Unreachable messages:

- **Administration** - The packet was dropped by the device due to a filter or ACL configured on the device.
- **Fragmentation-needed** - The packet has the Do not Fragment bit set in the IP Flag field, but the device cannot forward the packet without fragmenting it.
- **Host** - The destination network or subnet of the packet is directly connected to the device, but the host specified in the destination IP address of the packet is not on the network.
- **Network** - The device cannot reach the network specified in the destination IP address of the packet.
- **Port** - The destination host does not have the destination TCP or UDP port specified in the packet. In this case, the host sends the ICMP Port Unreachable message to the device, which in turn sends the message to the host that sent the packet.
- **Protocol** - The TCP or UDP protocol on the destination host is not running. This message is different from the Port Unreachable message, which indicates that the protocol is running on the host but the requested protocol port is unavailable.
- **Source-route-failure** - The device received a source-routed packet but cannot locate the next-hop IP address indicated in the packet's Source-Route option.

You can disable the device from sending these types of ICMP messages on an individual basis.

> NOTE
> Disabling an ICMP Unreachable message type does not change the device's ability to forward packets. Disabling ICMP Unreachable messages prevents the device from generating or forwarding the Unreachable messages.

To disable all ICMP Unreachable messages, enter the following command.

```
device(config)# no ip icmp unreachable
```

Syntax: [no] ip icmp unreachable [**network** | **host** | **protocol** | **administration** | **fragmentation-needed** | **port** | **source-route-fail**]

- If you enter the command without specifying a message type (as in the example above), all types of ICMP Unreachable messages listed above are disabled. If you want to disable only specific types of ICMP Unreachable messages, you can specify the message type. To disable more than one type of ICMP message, enter the **no ip icmp unreachable** command for each messages type.
- The **network** parameter disables ICMP Network Unreachable messages.
- The **host** parameter disables ICMP Host Unreachable messages.
- The **protocol** parameter disables ICMP Protocol Unreachable messages.
- The **administration** parameter disables ICMP Unreachable (caused by Administration action) messages.
- The **fragmentation-needed** parameter disables ICMP Fragmentation-Needed But Do not-Fragment Bit Set messages.
- The **port** parameter disables ICMP Port Unreachable messages.
- The **source-route-fail** parameter disables ICMP Unreachable (caused by Source-Route-Failure) messages.

To disable ICMP Host Unreachable messages and ICMP Network Unreachable messages but leave the other types of ICMP Unreachable messages enabled, enter the following commands instead of the command shown above.

```
device(config)# no ip icmp unreachable host
device(config)# no ip icmp unreachable network
```

If you have disabled all ICMP Unreachable message types but want to re-enable certain types, you can do so by entering commands such as the following.

```
device(config)# ip icmp unreachable host
device(config)# ip icmp unreachable network
```

These commands re-enable ICMP Unreachable Host messages and ICMP Network Unreachable messages.

Disabling ICMP redirect messages

ICMP redirect messages can be disabled or re-enabled. By default, the Extreme device sends an ICMP redirect message to the source of a misdirected packet in addition to forwarding the packet to the appropriate router. You can disable ICMP redirect messages on a global basis or on an individual port basis.

> NOTE
> An unusually high receipt of multiple Internet Control Message Protocol (ICMP) Redirect packets that are used to change routing table entries in a short period of time may cause high CPU utilization. This can be avoided by configuring the maximum ICMP error message rate using **ip icmp max-err-msg-rate** command, 0 (minimum) to 5000 (maximum) error message per second. The default value is 400.The total error rate for all IP interfaces (SYSTEM) is 10,000 errors per second.

> NOTE
> The device forwards misdirected traffic to the appropriate router, even if you disable the redirect messages.

To disable ICMP redirect messages globally, enter the following command at the global CONFIG level of the CLI.

> NOTE
> The **ip icmp redirects** command is applicable to the MLX Series and XMR Series devices only.

```
device(config)# no ip icmp redirects
```

Syntax: [no] ip icmp redirects

To disable ICMP redirect messages on a specific interface, enter the following command at the configuration level for the interface.

```
device(config)# int e 3/11
device(config-if-e100-3/11)# no ip redirect
```

Syntax: [no] ip redirect

IP addressing Show and Clear commands

There is a full range of IP addressing show and clear commands. They are documented in the command reference, and listed here with descriptions.

TABLE 18 IP addressing Show commands in the command reference

Command	Description
show ip interface	Displays the IP address, status, and configuration for a specified Ethernet, loopback, or VE interface. You can also display a brief summary of such information for all interfaces.
show ip route	Displays IP route information.

TABLE 19 IP addressing Clear commands in the command reference

Command	Description
clear ip route	Clears a specified route or all IP routes in the IP routing tables.

IPv6 Addressing

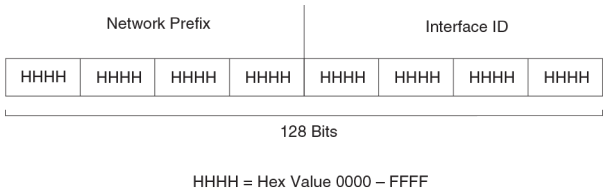
- IPv6 addressing overview....71
- Configuring an IPv6 address....72
- IPv6 management....72
- IPv6 neighbor discovery configuration....74
- Configuring an IPv6 domain name and server....80
- IPv6 MTU....80
- Information about IPv6 prefix list....81
- Displaying global IPv6 information....82
- Clearing global IPv6 information....84
- Prevention of DDoS attack on IPv6 subnet-router anycast address....85

IPv6 addressing overview

IPv6 increases the number of network address bits from 32 (IPv4) to 128 bits, which provides more unique IP addresses to support increasing number of network devices.

An IPv6 address comprise 8 fields of 16-bit hexadecimal values separated by colons (:). The following figure shows the IPv6 address format.

FIGURE 1 IPv6 address format



As shown in the above figure, HHHH is a 16-bit hexadecimal value, while H is a 4-bit hexadecimal value. The following is an example of an IPv6 address.

2001:0000:0000:0200:002D:D0FF:FE48:4672

Note that this IPv6 address includes hexadecimal fields of zeros. To make the address manageable, you can:

- Omit the leading zeros. For example, 2001:0:0:200:2D:D0FF:FE48:4672.
- Compress the successive groups of zeros at the beginning, middle, or end of an IPv6 address to two colons (::) once per address. For example, 2001::200:2D:D0FF:FE48:4672.

When specifying an IPv6 address in a command syntax, consider the following:

- You can use the two colons (::) only once in the address to represent the longest successive hexadecimal fields of zeros.
- The hexadecimal letters in IPv6 addresses are not case-sensitive.

As shown in Figure 1, the IPv6 network prefix is composed of the left-most bits of the address. As with an IPv4 address, you can specify the IPv6 prefix using the prefix/prefix-length format, where the following applies.

The prefix parameter is specified as 16-bit hexadecimal values separated by a colon.

The prefix-length parameter is specified as a decimal value that indicates the network portion of the IPV6 address.

The following is an example of an IPv6 prefix.

2001:DB8:49EA:D088::/64

Configuring an IPv6 address

You can configure global and link-local IPv6 addresses at the interface level.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode.

```
device(config)# interface ethernet 3/1
device(config)# interface ethernet 0/1
```

3. Configure a global IPv6 address for the interface.

```
device(config-if-e1000-3/1)# ipv6 address 2001:DB8:12D:1300:240:D0FF:FE48:4672:/64
device(conf-if-eth-0/1)# ipv6 address 2001:DB8:12D:1300:240:D0FF:FE48:4672/64
```

4. Configure a link-local IPv6 address for the interface. Link-local addresses are not forwarded outside the local network.

```
device(config-if-e1000-3/1)# ipv6 address FE80::240:D0FF:FE48:4672 link-local
device(config-if-eth-0/1)# ipv6 address FE80::240:D0FF:FE48:4672 link-local
```

5. Return to Privileged EXEC mode.

```
device(config-if-e1000-3/1)# end
device(config-if-eth-0/1)# end
```

The following example configures both a global IPv6 address and a link-local IPv6 address for the Ethernet interface.

```
device# configure terminal
device(config)# ipv6 unicast-routing
device(config)# interface ethernet 3/1
device(config-if-e1000-3/1)# ipv6 address 2001:DB8:12D:1300:240:D0FF:FE48:4672/64
device(config-if-e1000-3/1)# ipv6 address FE80::240:D0FF:FE48:4672 link-local

device# configure terminal
device(config)# interface ethernet 3/1
device(config-if-fo-3/1)# ipv6 address 2001:DB8:12D:1300:240:D0FF:FE48:4672/64
device(config-if-fo-3/1)# ipv6 address FE80::240:D0FF:FE48:4672 link-local

device# configure terminal
device(config)# interface ethernet 0/1
device(config-if-eth-0/1)# ipv6 address 2001:DB8:12D:1300:240:D0FF:FE48:4672/64
device(config-if-eth-0/1)# ipv6 address FE80::240:D0FF:FE48:4672 link-local
(config-if-eth-0/1)# end
```

IPv6 management

On the management interface of the SLX-OS device, the IPv6 routing functionality is not enabled.

Configuring IPv6 management ACLs

When you enter the **ipv6 access-list** command, the SLX-OS device enters the IPv6 Access List configuration level, where you can access several commands for configuring IPv6 ACL entries. After configuring the ACL entries, you can apply them to network management access features such as Telnet, SSH, Web, and SNMP.

> NOTE
> Unlike IPv4, there is no distinction between standard and extended ACLs in IPv6.

```
device(config)#ipv6 access-list netw
device(config-ipv6-access-list-netw)#
```

Syntax: [no] ipv6 access-list *ACL-name*

The *ACL-name* variable specifies a name for the IPv6 ACL. An IPv6 ACL name cannot start with a numeral, for example, 1access. Also, an IPv4 ACL and an IPv6 ACL cannot share the same name.

Specifying an IPv6 SNMP trap receiver

You can specify an IPv6 host as a trap receiver to ensure that all SNMP traps sent by the device go to the same SNMP trap receiver or set of receivers. To configure an IPv6 SNMP host, follow these steps:

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Use the **snmp-server host** command to specify an IPv6 host as a trap receiver.

   ```
   device(config)# snmp-server host ipv6 2001:DB8:89::13
   ```

Secure Shell, SCP, and IPv6

Secure Shell (SSH) is a mechanism that allows secure remote access to management functions on the device. SSH provides a function similar to Telnet. You can log in to and configure a device using a publicly or commercially available SSH client program, just as you can with Telnet. However, unlike Telnet, which provides no security, SSH provides a secure, encrypted connection to the device.

For more information on SSH, refer to *Extreme NetIron Switching Configuration Guide*.

IPv6 Telnet

Telnet sessions can be established between an Extreme device to a remote IPv6 host, and from a remote IPv6 host to the device using IPv6 addresses.

The **telnet** command establishes a Telnet connection from an Extreme device to a remote IPv6 host using the console. Up to five read-access Telnet sessions are supported on the router at one time. Write-access through Telnet is limited to one session, and only one outgoing Telnet session is supported on the router at one time.

Use the **show telnet server status** command to view the status of the Telnet sever.

Establishing a Telnet session from an IPv6 host

To establish a Telnet session from an IPv6 host to the SLX-OS device, open your Telnet application and specify the IPv6 address of the device. The following example shows how to establish a Telnet session.

Use the **telnet** command to establish a Telnet session to an SLX-OS device.

```
device# telnet 2001:DB8:3de2:c37::6
```

IPv6 traceroute

Use the **traceroute** command to trace a path from the SLX-OS device to an IPv6 host.

```
device# traceroute ipv6 2001:DB8:349e:a384::34
```

The **traceroute** command displays trace route information for each hop as soon as the information is received. The traceroute requests display all responses of a minimum TTL of 1 second and a maximum TTL of 30 seconds. In addition, if there are multiple equal-cost routes to the destination, the device displays up to three responses.

Restricting Web management access

You can restrict Web management access to include only management functions on an SLX-OS device that is acting as an IPv6 host, or restrict access so that the SLX-OS host can be reached by a specified IPv6 device.

IPv6 neighbor discovery configuration

The neighbor discovery feature for IPv6 uses IPv6 ICMP messages to perform the following tasks:

- Determine the link-layer address of a neighbor on the same link.
- Verify that a neighbor is reachable.
- Track neighbor routers.

An IPv6 host is required to listen for and recognize the following addresses that identify itself:

- Link-local address.
- Assigned unicast address.
- Loopback address.
- All-nodes multicast address.
- Solicited-node multicast address.
- Multicast address to all other groups to which it belongs.

You can adjust the following IPv6 neighbor discovery features:

- Neighbor solicitation messages for duplicate address detection.
- Router advertisement messages:
  - Interval between router advertisement messages.
  - Value that indicates a router is advertised as a default router (for use by all nodes on a given link).
  - Prefixes advertised in router advertisement messages.
  - Flags for host stateful autoconfiguration.
- Amount of time during which an IPv6 node considers a remote node reachable (for use by all nodes on a given link).

- The time interval after which the IPv6 Neighbor Discovery cache is deleted or refreshed.

IPv6 neighbor discovery configuration notes

NOTE
For all solicitation and advertisement messages, SLX-OS uses seconds as the unit of measure instead of milliseconds.

- If you add a port to a port-based VLAN, and the port has IPv6 neighbor discovery configuration, the system cleans up the neighbor discovery configuration from the port and displays the following message on the console.

```
ND6 port config on the new member ports removed
```

- Neighbor discovery is not supported on tunnel interfaces.

Neighbor solicitation and advertisement messages

Neighbor solicitation and advertisement messages enable a node to determine the link-layer address of another node (neighbor) on the same link. (This function is similar to the function provided by the Address Resolution Protocol [ARP] in IPv4.) For example, node 1 on a link wants to determine the link-layer address of node 2 on the same link. To do so, node 1, the source node, multicasts a neighbor solicitation message. The neighbor solicitation message, which has a value of 135 in the Type field of the ICMP packet header, contains the following information:

- Source address: IPv6 address of node 1 interface that sends the message.
- Destination address: solicited-node multicast address (FF02:0:0:0:0:1:FF00::/104) that corresponds the IPv6 address of node 2.
- Link-layer address of node 1.
- A query for the link-layer address of node 2.

After receiving the neighbor solicitation message from node 1, node 2 replies by sending a neighbor advertisement message, which has a value of 136 in the Type field of the ICMP packet header. The neighbor solicitation message contains the following information:

- Source address: IPv6 address of the node 2 interface that sends the message.
- Destination address: IPv6 address of node 1.
- Link-layer address of node 2.

After node 1 receives the neighbor advertisement message from node 2, nodes 1 and 2 can now exchange packets on the link.

After the link-layer address of node 2 is determined, node 1 can send neighbor solicitation messages to node 2 to verify that it is reachable. Also, nodes 1, 2, or any other node on the same link can send a neighbor advertisement message to the all-nodes multicast address (FF02::1) if there is a change in their link-layer address.

Router advertisement and solicitation messages

Router advertisement and solicitation messages enable a node on a link to discover the routers on the same link.

Each configured router interface on a link sends out a router advertisement message, which has a value of 134 in the Type field of the ICMP packet header, periodically to the all-nodes link-local multicast address (FF02::1).

A configured router interface can also send a router advertisement message in response to a router solicitation message from a node on the same link. This message is sent to the unicast IPv6 address of the node that sent the router solicitation message.

At system startup, a host on a link sends a router solicitation message to the all-routers multicast address (FF01). Sending a router solicitation message, which has a value of 133 in the Type field of the ICMP packet header, enables the host to automatically configure its IPv6 address immediately instead of awaiting the next periodic router advertisement message.

Because a host at system startup typically does not have a unicast IPv6 address, the source address in the router solicitation message is usually the unspecified IPv6 address (0:0:0:0:0:0:0:0). If the host has a unicast IPv6 address, the source address is the unicast IPv6 address of the host interface sending the router solicitation message.

Neighbor redirect messages

After forwarding a packet, by default, a router can send a neighbor redirect message to a host to inform it of a better first-hop router. The host receiving the neighbor redirect message will then readdress the packet to the better router.

A router sends a neighbor redirect message only for unicast packets, only to the originating node, and to be processed by the node.

A neighbor redirect message has a value of 137 in the Type field of the ICMP packet header.

Duplicate Address Detection (DAD)

Although the stateless auto configuration feature assigns the 64-bit interface ID portion of an IPv6 address using the MAC address of the host's NIC, duplicate MAC addresses can occur. Therefore, the duplicate address detection feature verifies that a unicast IPv6 address is unique before it is assigned to a host interface by the stateless auto configuration feature. Duplicate address detection verifies that a unicast IPv6 address is unique.

If duplicate address detection identifies a duplicate unicast IPv6 address, the address is not used. If the duplicate address is the link-local address of the host interface, the interface stops processing IPv6 packets.

In the DAD NS message, the source address field in the IPv6 header is set to the unspecified address (::). The address being queried for duplication cannot be used until it is determined that there are no duplicates. In the neighbor advertisement (NA) reply to a DAD NS message, the destination address in the IPv6 header is set to the link-local all-nodes multicast address (FF02::1). The Solicited flag in the NA message is set to 0. Because the sender of the DAD NS message is not using the desired IP address, it cannot receive unicast NA messages. Therefore, the NA message is multicast.

Upon receipt of the multicast NA message with the target address field set to the IP address for which duplication is being detected, the node disables the use of the duplicate IP address on the interface. If the node does not receive an NA message that defends the use of the address, it initializes the address on the interface.

> **NOTE**
> Duplicate Address Detection (DAD) is not currently supported with IPv6 tunnels. Make sure tunnel endpoints do not have duplicate IP addresses.

You can configure the following neighbor solicitation message parameters that affect duplicate address detection while it verifies that a tentative unicast IPv6 address is unique:

- The number of consecutive neighbor solicitation messages that duplicate address detection sends on an interface. By default, duplicate address detection sends three neighbor solicitation messages without any follow-up messages.
- The number of consecutive neighbor solicitation messages that duplicate address detection sends on an interface. By default, duplicate address detection sends two neighbor solicitation messages without any follow-up messages.
- The interval in seconds at which duplicate address detection sends a neighbor solicitation message on an interface. By default, duplicate address detection sends a neighbor solicitation message every 1000 milliseconds.

Configuring IPv6 static neighbor entries

In some cases a neighbor cannot be reached by means of Neighbor Discovery. To resolve this you can add a static entry to the ND cache, causing a neighbor to reachable at all times. (A static IPv6 ND entry is like a static IPv4 ARP entry.)

For example, use the **ipv6 neighbor** command in interface subtype configuration mode to add a static entry for a neighbor with IPv6 address 2001:db8:2678::2 and link-layer address 0000.002b.8641, reachable through interface te 3/0/1.

For example, use the **ipv6 neighbor** command in interface subtype configuration mode to add a static entry for a neighbor with IPv6 address 2001:db8:2678::2 and link-layer address 0000.002b.8641, reachable through interface ethernet 3/5.

```
switch(config-if-te-3/0/1)# ipv6 neighbor 2001:db8:2678::2 0000.002b.8641
```

```
switch(config-if-eth-0/1)# ipv6 neighbor 2001:db8:2678::2 0000.002b.8641
```

IPv6 router advertisement parameters

You can adjust the following parameters for router advertisement messages:

- The interval (in seconds) at which an interface sends router advertisement messages. By default, an interface sends a router advertisement message every 200 seconds.
- The interval (in seconds) at which an interface sends router advertisement messages. By default, an interface sends a router advertisement message every 200 to 600 seconds.
- The "router lifetime" value, which is included in router advertisements sent from a particular interface. The value (in seconds) indicates if the router is advertised as a default router on this interface. If you set the value of this parameter to 0, the router is not advertised as a default router on an interface. If you set this parameter to a value that is not 0, the router is advertised as a default router on this interface. By default, the router lifetime value included in router advertisement messages sent from an interface is 1800 seconds.
- The hop limit to be advertised in the router advertisement.

Setting IPv6 router advertisement parameters

When adjusting these parameter settings, Extreme recommends that the interval between router advertisement transmission be less than or equal to the router lifetime value if the router is advertised as a default route. To configure the IPv6 router advertisement parameters, perform the following tasks.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Use the **interface** command to enter the interface configuration mode.

   ```
   device# interface ethernet 0/1
   ```

3. Use the **ipv6 nd ra-lifetime** command to configure the router advertisement lifetime.

   ```
   device(config-if-eth-0/1)# ipv6 nd ra-lifetime 1900
   ```

4. Use the **ipv6 nd ra-interval** command to set a maximum interval range and minimum interval at which RA messages are sent.

   ```
   device(config-if-eth-0/1)# ipv6 nd ra-interval 1200 min 400
   ```

5. Use the **ipv6 nd ra-lifetime** command to set the RA message lifetime.

   ```
   device(config-if-eth-0/1)# ipv6 nd ra-lifetime 1900
   ```

6. Use the **ipv6 nd hoplimit** command to specify a nondefault hop limit.

```
device(config-if-eth-0/1)# ipv6 nd hoplimit 32
```

7. Use the **ipv6 nd mtu** command to specify a nondefault MTU.

```
device(config-if-eth-0/1)# ipv6 nd mtu 2400
```

Flags in IPv6 router advertisement messages

An IPv6 router advertisement message can include the following flags:

- Managed Address Configuration--This flag indicates to hosts on a local link if they should use the stateful autoconfiguration feature to get IPv6 addresses for their interfaces. If the flag is set, the hosts use stateful autoconfiguration to get addresses as well as non-IPv6-address information. If the flag is not set, the hosts do not use stateful autoconfiguration to get addresses and if the hosts can get non-IPv6-address information from stateful autoconfiguration is determined by the setting of the Other Stateful Configuration flag.
- Other Stateful Configuration--This flag indicates to hosts on a local link if they can get non-IPv6 address autoconfiguration information. If the flag is set, the hosts can use stateful autoconfiguration to get non-IPv6-address information.

  NOTE
  When determining if hosts can use stateful autoconfiguration to get non-IPv6-address information, a set Managed Address Configuration flag overrides an unset Other Stateful Configuration flag. In this situation, the hosts can obtain nonaddress information. However, if the Managed Address Configuration flag is not set and the Other Stateful Configuration flag is set, then the setting of the Other Stateful Configuration flag is used.

Setting flags in IPv6 router advertisement messages

By default, the managed address configuration and other stateful configuration flags are not set in router advertisement messages.

1. Enter global configuration mode.

```
device# configuration terminal
```

2. Use the **interface** command to enter interface configuration mode.

```
device (config)# interface ethernet 0/1
```

3. Use the **ipv6 nd managed-config-flag** command to configure the managed address configuration flag.

```
device(conf-if-eth-0/1)# ipv6 nd managed-config-flag
```

4. Use the **ipv6 nd other-config-flag** command to configure the other stateful configuration flag.

```
device(conf-if-eth-0/1)# ipv6 nd other-config-flag
```

   To remove either flag from router advertisement messages sent on an interface, use the **no** form of the corresponding command.

Sending or suppressing IPV6 router advertisements and addresses

When IPV6 address is configured on an Ethernet interface, the interface sends router advertisement (RA) messages by default.

1. Enter interface configuration mode.

```
device# configure terminal
```

2. Use the **ipv6 nd global-suppress-ra** to enable the sending of RA messages on all interfaces.

```
device(conf)# ipv6 nd global-suppress-ra
```

NOTE
The interface specific command overrides this global configuration.

3. Use the **ipv6 nd send-ra** command to allow the user to keep some selected interfaces sending RA messages when ipv6 nd global-suppress-ra command is set.

```
device (conf-if-eth-0/1)# ipv6 nd send-ra
```

4. Use the **ipv6 nd suppress-ra** command to allow the user to selectively stop interfaces from sending RA messages when ipv6 nd global-suppress-ra command is not set.

```
device (conf-if-eth-0/1)# ipv6 nd suppress-ra
```

5. Use the **ipv6 nd address** command to allow a user to specify an address to be suppressed or all the addresses to be suppressed on a given interface.

```
device (conf-if-eth-0/1)# ipv6 nd address 2001:DB8:12D:1300:240:D0FF:FE48:4672 suppress
```

IPv6 router advertisement preference support

IPv6 router advertisement (RA) preference enables IPv6 RA messages to communicate default router preferences from IPv6 routers to IPv6 hosts in network topologies where the host has multiple routers on its Default Router List. This improves the ability of the IPv6 hosts to select an appropriate router for an off-link destination.

Configuring IPv6 RA preference

Configuring IPv6 RA preference

If IPv6 unicast routing is enabled on an Ethernet interface, by default, this interface sends IPv6 router advertisement messages. The IPv6 router sets the preference field based on the configured value on IPv6 RA and sends it periodically to the IPv6 host or as a response to the router solicitations.

1. Enter global configuration mode.

```
device# configuration terminal
```

2. Use the **ipv6 nd router-preference** command to configure IPv6 RA preference for the IPv6 router.

```
device(config)# interface ethernet 1/2
device(config-if-e1000-1/2)# ipv6 nd router-preference low

device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# ipv6 nd router-preference medium
```

Reachable time for remote IPv6 nodes

The router advertisement messages sent by a router interface include the duration of time specified so that nodes on a link use the same reachable time duration. By default, the messages include a default value of 0.

Extreme recommends configuring a longer reachable time duration, because a short duration causes the IPv6 network devices to process the information at a greater frequency.

Configuring reachable time for remote IPv6 nodes

You can configure the duration (in seconds) that a router considers a remote IPv6 node reachable. By default, a router interface uses 30 seconds.

1. Enter global configuration mode.

```
device# configuration terminal
```

2. Use the **ipv6 nd reachable-time** command to configure the reachable time.

```
device(config)# interface ethernet 0/1
device (conf-if-eth-0/1)# ipv6 nd reachable-time 600
```

> NOTE
> The actual reachable time will be from 0.5 to 1.5 times the configured or default value.

Configuring an IPv6 domain name and server

You can configure a maximum number of 4 DNS Recursive Server addresses and corresponding lifetime . If you configure at least one DNS Recursive Server on the interface, it overrides all the configurations of the DNS Recursive Server addresses at system level for this interface. Configuring DNS attributes take effect in the next scheduled RA as specified by the interval value.

1. Use the **ipv6 nd ra-dns-server** command to configure an IPv6 domain name and server.

```
device(conf-if-eth-0/1)# ipv6 nd ra-dns-server 2001:DB8:0:ee44::1 lifetime 200
```

2. Use the **ipv6 nd ra-domain-name** command to configure a DNS Recursive domain name.

```
device(conf-if-eth-0/1)# ipv6 nd ra-domain-name 2001:DC8:200::3 lifetime 200
```

IPv6 MTU

The IPv6 maximum transmission unit (MTU) is the maximum length of an IPv6 packet that can be transmitted on a particular interface. If an IPv6 packet is longer than an MTU, the host that originated the packet fragments the packet and transmits its contents in multiple packets that are shorter than the configured MTU.

The hardware only supports three MTU profiles, including the default value of 1500 and two user-defined values. If the limit is reached, the following error is displayed:

```
%Error: Maximum limit of allowed different IP MTUs reached.
```

The hardware supports only one MTU value at the interface.

MTU supported values are 1300, 1500, 9194 and default value is 1500. The previous lower value will be selected when you configure a value other than the supported .

Hardware supports only one MTU value at the interface, though CLI allows different value for v4 and v6. Last configured value will be programmed into hardware. It is recommended to configure same MTU value for v4 and v6 in case of dual stack.

By default, the MTU size is 1548 bytes. The maximum Ethernet MTU size is 9216.

Changing IPv6 MTU

To change the IPv6 MTU value, perform the following steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Use the **ip mtu** command to change the IP MTU globally.

   ```
   device(config)# ip mtu 2000
   ```

   The **ip mtu** command configures both IPv4 and IPv6 MTU.

3. Use the **interface** command to enter the interface Ethernet configuration mode.

   ```
   device(config)# interface ethernet 0/1
   ```

4. Change the MTU.

   ```
   device(conf-if-eth-0/1)# ip mtu 1300
   device(conf-if-eth-0/1)# ipv6 mtu 1300
   ```

   The configured interface value takes precedence over the configured global MTU value.

   You cannot configure IPv6 MTU globally.

Information about IPv6 prefix list

An IPv6 prefix list comprises one or more conditional statements that pose an action (permit or deny) if a route matches a specified prefix. In prefix lists with multiple statements, you can specify a sequence number for each statement. The specified sequence number determines the order in which the statement appears in the prefix.

You can configure an IPv6 prefix list on a global basis and use it as input to other commands or processes, such as route aggregation, route redistribution, route distribution, route maps, and so on. When a device sends or receives an IPv6 route, it applies the statements within the IPv6 prefix list in their order of appearance to the packet. When a match occurs, the device takes the specified action (permit or deny the packet) and stops further comparison for that route.

You can use permit statements in the prefix list to specify the route that you want to send to the other feature. If you use deny statements, the route specified by the deny statements is not supplied to the other feature.

A device supports IPv6 prefix lists, which you can use for basic route filtering. You can configure up to 100 IPv6 prefix lists.
You must specify the ipv6-prefix parameter in hexadecimal using 16-bit values between colons as documented in RFC 4291. You must specify the prefix-length parameter as a decimal value. A slash mark (/) must follow the ipv6-prefix parameter and precede the prefix-length parameter.

Configuring an IPv6 prefix list

To configure an IPv6 prefix list for basic traffic filtering, perform the following tasks.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Use the **ipv6 prefix list** command to configure an IPv6 prefix list.

```
(config)# ipv6 prefix-list routesfor2001 permit 2001:db8::/32
```

3. Use the **ipv6 prefix-list** command configure an IPv6 prefix list.

```
(config)# ipv6 prefix-list prefix-filter-1 permit FE80::/10 ge 25 le 25
```

   The **ge** *ge-value* or **le** *le-value* that you specify must meet the following condition for prefix-length:

   ge-value <= le-value <= 128

   If you do not specify **ge** *ge-value* or **le** *le-value*, the prefix list matches only on the exact prefix you specify with the ipv6-prefix/prefix-length parameter.

4. Use the **ipv6 router rip** command to enable RIPng globally.

```
(config)# ipv6 router rip
```

5. Use the **distribute-list** command to configure an IPv6 prefix list and use it as input to the RIPng.

```
(config-ripng-router)# distribute-list prefix-list routesfor2001 out ethernet 3/1
```

   These commands permit the inclusion of routes with the IPv6 prefix 2001:db8::/32 in RIPng routing updates sent from Ethernet interface 3/1.

6. To delete a prefix-list entry use the **no ipv6 prefix-list** command.

```
(config)# no ipv6 prefix-list 2001:db8::/32
```

Displaying prefix list information

To display the IPv6 prefix lists configured on a device, use the **show ipv6 prefix-lists** command. Extreme

To display the IPv6 prefix lists configured on an SLX-OS device, use the **show ipv6 prefix-list** command

```
device# show ipv6 prefix-lists
routesfor2001
 ipv6 prefix-list routesfor2001
   seq 5 permit 2001::/16
   seq 10 permit 2001:db8::/32

device# show ipv6 prefix-list routesfor2001
ipv6 prefix-list routesfor2001: 2 entries
     seq 5 permit 2001::/16
     seq 10 permit 2001:db8::/32
```

Displaying global IPv6 information

You can display output for the following global IPv6 parameters:

- IPv6 interfaces

- IPv6 neighbors
- IPv6 route table

Displaying IPv6 interface information

Use the **show ipv6 interface** command to display IPv6 interface information.

```
device# show ipv6 interface brief

Interface                         Vrf                     Status        Protocol
         IPv6-Address
================================================================================
Loopback 1                        default-vrf             up            up
         3911::4/128
Ethernet 0/1                      default-vrf             up            up
         3002::4/64

Ve 2                              default-vrf             up            down
         a1a1:0:2::2/64
Ve 3                              default-vrf             up            down
         a1a1:0:3::2/64
Ve 4                              default-vrf             up            down
         a1a1:0:4::2/64
```

Displaying IPv6 neighbor information

Use the **show ipv6 neighbor** command to display the IPv6 neighbor table.

```
device# show ipv6 neighbor
Address                                       Mac-address     Interface     MacResolved
Age         Type
-----------------------------------------------------------------------------------------------------
-------
a1a1:0:3::1                                   609c.9f02.1f15  Ve 3          no
10:25:32    Dynamic
fe80::21b:edff:fe9f:1900                      001b.ed9f.1900  Eth 0/1       yes
00:02:36    Dynamic
fe80::629c:9fff:fe02:1f15                     609c.9f02.1f15  Ve 2          no
10:25:32    Dynamic
fe80::629c:9fff:fe02:1f15                     609c.9f02.1f15  Ve 3          no
10:25:42    Dynamic
fe80::629c:9fff:fe02:1f15                     609c.9f02.1f15  Ve 4          no
10:32:04    Dynamic
fe80::629c:9fff:fe02:1f15                     609c.9f02.1f15  Ve 5          no
10:32:05    Dynamic
```

Displaying the IPv6 route table

Use the **show ipv6 route** command to display the IPv6 route table.

```
device# show ipv6 route
IPv6 Routing Table for VRF "default-vrf"
Total number of IPv6 routes: 5
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

3002::/64, attached
    *via ::, Eth 1/1, [0/0], 10h19m, direct, tag 0
3002::4/128, attached
    *via ::, Eth 1/1, [0/0], 10h19m, local, tag 0
3911::4/128, attached
    *via ::, Lo 1, [0/0], 12h6m, direct, tag 0
fe80::/10, attached
    *via ::, , [0/0], 12h6m, local, tag 0
ff00::/8, attached
    *via ::, Null0, [0/0], 12h6m, local, tag 0
```

Clearing global IPv6 information

You can clear the following global IPv6 information from an SLX-OS device:

- Entries from the IPv6 neighbor table
- IPv6 routes from the IPv6 route table

Clearing IPv6 neighbor information

You can remove all entries from the IPv6 neighbor table or to specify an entry based on the IPV6 prefix, IPv6 address, and interface type.

Use the **clear ipv6 neighbor** command to remove the IPv6 neighbor table entries.

```
device# clear ipv6 neighbor ethernet 1/3
```

```
device# clear ipv6 neighbor 2000:7838::1 force-delete
```

Clearing IPv6 routes from the IPv6 route table

You can clear all IPv6 routes or only those routes associated with a particular IPv6 prefix from the IPv6 route table and reset the routes.

Use the **clear ipv6 route** command to clear IPv6 routes.

```
device# clear ipv6 route 2000:7838::/32
```

Use the **all** keyword to clear all Ipv6 routes. Use the **slot** keyword to clear IPv6 route on a specific slot (LP).

Prevention of DDoS attack on IPv6 subnet-router anycast address

To prevent Distributed Denial of Service (DDoS) attack, you can disable CPU processing (by enabling a hardware drop) of packets destined for IPv6 subnet-router anycast addresses.

RFC 4291 requires that all devices support IPv6 subnet-router anycast address which is defined by an all-zero subnet prefix. Traffic sent to an IPv6 subnet-router anycast address is delivered to one device on the subnet and devices are required to support subnet-router anycast addresses for the subnets to which they have interfaces. The IPv6 subnet-router anycast address is intended for use by applications in which a node needs to communicate with any one of the set of devices. However, it creates a vulnerability whereby a DDoS attack may be launched by sending traffic to the IPv6 subnet-router anycast address.

To prevent DDoS attack, you can block traffic destined for the IPv6 subnet-router anycast address by using the **ipv6 subnet-zero drop** command. When this command is enabled, the hardware is programmed to drop all IPv6 subnet-router anycast address traffic.

Enabling hardware drop of IPv6 subnet-router anycast address traffic

Enabling hardware drop of IPv6 subnet-router anycast address traffic, programs the hardware to drop IPv6 traffic with the all-zero subnet prefix.

By default, packets destined for IPv6 subnet-router anycast addresses are allowed. Perform the following steps to drop all IPv6 subnet-router anycast address traffic.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable hardware drop of traffic destined for IPv6 subnet-router anycast addresses.

   ```
   device(config)# ipv6 subnet-zero drop
   ```

   Use the **no ipv6 subnet-zero drop** command, when you want to restore the default configuration and allow IPv6 subnet-router anycast address traffic.

The following example enables hardware drop of IPv6 subnet-router anycast address traffic.

```
device# configure terminal
device(config)# ipv6 subnet-zero drop
```

IPv4 Static Routing

- Overview of static routing....87
- Configuring a basic IP static route....89
- Adding metrics to a static route....90
- Naming an IP static route....92
- Removing a name or a static route....92
- Configuring a physical interface as next hop....92
- Configuring a virtual interface as next hop....94
- Configuring a static route with a VRF as next hop....95
- Configuring an MPLS next hop for an IPv4 static route....96
- Configuring a static route for use with a route map....97
- Configuring a null route....98
- Configuring a default static route....100
- Resolving a static route using other static routes....101
- Resolving the next hop through a protocol....102
- Static routes between VRFs....103
- Configuring load sharing and redundancy....108
- Displaying IPv4 static routes....111

Overview of static routing

Static routes are manually configured entries in the IP routing table.

The IP route table can receive routes from several sources, including static routes. Other route sources include directly connected networks, RIP, OSPF, and BGP4 protocols.

The IP route table can receive routes from several sources, including static routes. Other route sources include directly connected networks, RIP, OSPF, BGP4, and ISIS protocols.

The IP route table can receive routes from several sources, including static routes. Other route sources include directly connected networks, OSPF, BGP4, and ISIS protocols.

The IP route table can receive routes from several sources, including static routes. Other route sources include directly connected networks, OSPF and BGP4 protocols.

Static routes can be used to specify desired routes, backup routes, or routes of last resort. Static routing can help provide load balancing and can use routing information learned from other protocols.

In setting up static routes, you can specify several types of destinations:

- Destination network, using an IP address and network mask or prefix length
- Default network route
- Next-hop router
- Next-hop network protocol type
- Ethernet interface, typically used for directly attached destination networks
- Virtual interface
- Null interface
- Destination network, using an IP address and prefix length

- Default network route
- Next-hop router
- Ethernet interface, typically used for directly attached destination networks
- Virtual interface
- Null interface

You can influence the preference a route is given in the following ways:

- By setting a route metric higher than the default metric
- By giving the route an administrative distance
- By specifying a route tag for use with a route map.

Static routes can be configured to serve as any of the following:

- Default routes
- Primary routes
- Backup routes
- Null routes for intentionally dropping traffic when the desired connection fails
- Alternative routes to the same destination to help load balance traffic.

Static route states follow port states

IP static routes remain in the IP route table only as long as the port or virtual interface used by the route is available and the next-hop IP address is valid; otherwise, the software removes the static route from the IP route table. If the port or virtual routing interface becomes available again later and the next-hop is valid, the software adds the route back to the route table.

This feature allows the router to adjust to changes in network topology. The router does not continue trying to use routes on unavailable paths but instead uses routes only when their paths are available.

In the following example, a static route is configured on Switch A. The route configuration is shown following the figure.

FIGURE 2 Example of static route

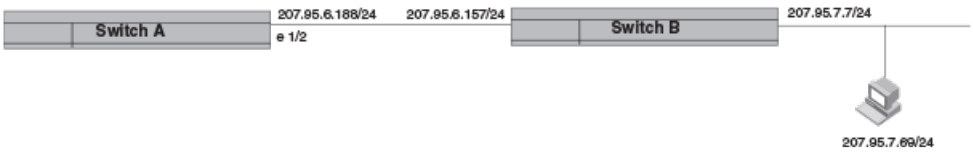


FIGURE 3 Example of static route

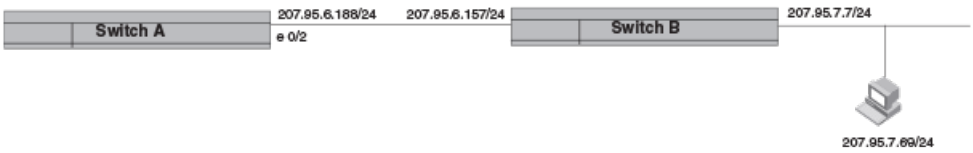
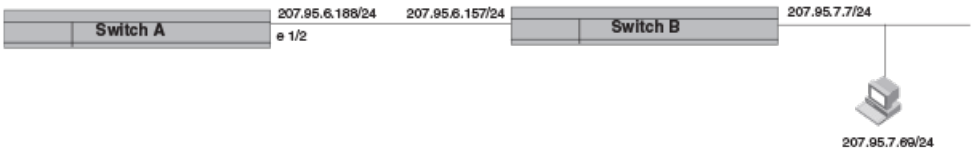


FIGURE 4 Example of static route



The following command configures a static route to 207.95.7.0 destinations, using 207.95.6.157 as the next-hop gateway.

```
device(config)# ip route 207.95.7.0/24 207.95.6.157
```

When you configure a static IP route, you specify the destination address for the route and the next-hop gateway or Layer 3 interface through which the Layer 3 device can reach the route. The device adds the route to the IP route table. In this case, Switch A knows that 207.95.6.157 is reachable through port 1/2, and also assumes that local interfaces within that subnet are on the same port. Switch A deduces that IP interface 207.95.7.7 is also on port 1/2.

When you configure a static IP route, you specify the destination address for the route and the next-hop gateway or Layer 3 interface through which the Layer 3 device can reach the route. The device adds the route to the IP route table. In this case, Switch A knows that 207.95.6.157 is reachable through port 0/2, and also assumes that local interfaces within that subnet are on the same port. Switch A deduces that IP interface 207.95.7.7 is also on port 0/2.

When you configure a static IP route, you specify the destination address for the route and the next-hop gateway or Layer 3 interface through which the Layer 3 device can reach the route. The device adds the route to the IP route table. In this case, Switch A knows that 207.95.6.157 is reachable through port 1/2, and also assumes that local interfaces within that subnet are on the same port. Switch A deduces that IP interface 207.95.7.7 is also on port 1/2.

The software automatically removes a static IP route from the IP route table if the port used by that route becomes unavailable or the IP address is not valid. When the port becomes available again, the software automatically re-adds the route to the IP route table.

Configuring a basic IP static route

To configure a basic IP static route, perform these steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the RBridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Enter the IP address and prefix length, or enter the IP address and network mask for the route destination network. On the same command line, enter the IP address for the next hop.

   ```
   device(config)# ip route 10.0.0.0 255.0.0.0 10.1.1.1
   ```

   This example configures an IP static route with a destination network address of 10.0.0.0, a destination mask of 255.0.0.0, and a next hop address of 10.1.1.1.

   NOTE
   In the network mask, "1's" are significant bits, and "0's" allow any value. The mask 255.255.255.0 matches all hosts within the Class C subnet address specified in the destination IP address. You can use "/24" as the equivalent address prefix.

4. Enter the IP address and prefix length for the route destination network. On the same command line, enter the IP address for the next hop.

```
device(config)# ip route 10.0.0.0/24 10.1.1.1
```

This example configures an IP static route with a destination network address of 10.0.0.0, a prefix length of /24, and a next hop address of 10.1.1.1.

> **NOTE**
> Prefix lengths must be used as part of the address. Network masks cannot be used. The prefix length of /8 is equivalent to a network mask of 255.0.0.0. The prefix length of /24 (equivalent to the mask 255.255.255.0) matches all hosts within the Class C subnet address specified in the destination IP address.

5. Enter the IP address and prefix length for the route destination network. On the same command line, enter the IP address for the next hop.

```
device(config-rbridge-id-30)# ip route 10.0.0.0/8 10.1.1.1
```

This example configures an IP static route with a destination network address of 10.0.0.0, a prefix length of /8, and a next hop address of 10.1.1.1.

> **NOTE**
> Prefix lengths must be used as part of the address. Network masks cannot be used. The prefix length of /8 is equivalent to a network mask of 255.0.0.0. The prefix length of /24 (equivalent to the mask 255.255.255.0) matches all hosts within the Class C subnet address specified in the destination IP address.

The following example configures an IP static route with a destination network address of 10.0.0.0, a prefix length of 24 bits, and a next hop address of 10.1.1.1.

```
device# configure terminal
device(config)# ip route 10.0.0.0/24 10.1.1.1
```

The following example configures an IP static route with a destination network address of 10.0.0.0, a prefix length of 24 bits, and a next hop address of 10.1.1.1.

```
device# configure terminal
device(config)# ip route 10.0.0.0/24 10.1.1.1
```

The following example configures an IP static route on Rbridge ID 30 with a destination network address of 10.0.0.0, a prefix length of 24 bits, and a next hop address of 10.1.1.1.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 10.0.0.0/24 10.1.1.1
```

Adding metrics to a static route

You can influence route preference by adding a cost metric or an administrative distance to a static route.

Follow these steps to create an IP static route with cost metrics.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Designate the route destination and next hop, and add a route priority parameter.

Option	Description
Cost metric	The value is compared to the metric for other static routes in the IPv4 route table to the same destination. Two or more routes to the same destination with the same metric load share traffic to the destination. The value may range from 1 through 16. The default is 1. A route with a cost of 16 is considered unreachable.
Administrative distance	This value is compared to the administrative distance of all routes to the same destination. Static routes by default take precedence over learned protocol routes. However, to give a static route a lower priority than a dynamic route, give the static route the higher administrative distance. The value is preceded by the keyword **distance** and can range from 1 to 255. The default is 1. A value of 255 is considered unreachable.

Option	Description
Cost metric	The value is compared to the metric for other static routes in the IPv4 route table to the same destination. Two or more routes to the same destination with the same metric load share traffic to the destination. The value may range from 1 through 16. The default is 1. A route with a cost of 16 is considered unreachable.
Administrative distance	This value is compared to the administrative distance of all routes to the same destination. Static routes by default take precedence over learned protocol routes. However, to give a static route a lower priority than a dynamic route, give the static route the higher administrative distance. The value is preceded by the keyword **distance** and can range from 1 to 254. The default is 1. A value of 255 is considered unreachable.

```
device(config)# ip route 10.128.2.71/24 10.111.10.1 distance 10

device(config-rbridge-id-30)# ip route 10.128.2.71/24 10.111.10.1 distance 10
```

This example configures a static route with an administrative distance of 10.

NOTE
The device replaces a static route if it receives a route to the same destination with a lower administrative distance.

```
device(config)# ip route 10.128.2.69/24 10.111.10.1 2

device(config-rbridge-id-30)# ip route 10.128.2.69/24 10.111.10.1 2
```

This example configures a static route with a metric of 2.

The following example configures a static route to destinations with an IP address beginning with 10.0.0.0. The route uses IP address 10.111.10.1 as the next hop. The static route is assigned an administrative distance of 3.

```
device# configure terminal
device(config)# ip route 10.0.0.0/24 10.111.10.1 distance 3

device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 10.0.0.0/24 10.111.10.1 distance 3
```

Naming an IP static route

IPv4 static route names are optional and non-unique. You can give a group of static routes the same name to help identify them.

To configure an IP static route with a name, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the destination IP address and mask or prefix length followed by the IP address of the next hop. On the same command line, enter the keyword name followed by the identifying ASCII string.

```
device(config)# ip route 10.22.22.22 255.255.255.0 10.1.1.1 name corporate
```

The following example creates a static route to IP destination network addresses beginning with 10.22.22.22 through the next-hop address 10.1.1.1. The route is given the non-unique name "corporate."

```
device# configure terminal
device(config)# ip route 10.22.22.22 255.255.255.0 10.1.1.1 name corporate
```

Removing a name or a static route

When an IP route has a name, the **no** form of the full **ip route** command removes the name. Use the **no** form of the command a second time to remove the route.

1. Enter configuration mode.

```
device# configure terminal
```

2. Enter **no ip route** followed by the full route designation.

```
device# configure terminal
device(config)# no ip route 10.22.22.22 255.255.255.255 10.1.1.1 name xyz
```

   This example removes only the name of the route.

3. If necessary, repeat the **no ip route** command with the full route designation.

```
device(config)# no ip route 10.22.22.22 255.255.255.255 10.1.1.1
```

   This example repeats the previous route. Because the route has no name, the command removes the designated static route.

The following example removes the name of the designated static route, removes the route, and saves the change to the IP routing table.

```
device# configure terminal
device(config)# no ip route 10.22.22.22 255.255.255.255 10.1.1.1 name xyz
device(config)# no ip route 10.22.22.22 255.255.255.255 10.1.1.1
```

Configuring a physical interface as next hop

The interface you use for the static route's next hop must have at least one IP address configured on it. The address does not need to be in the same subnet as the destination network.

> NOTE
> You cannot add an interface-based static route to a network if there is already a static route of any type with the same metric you specify for the interface-based route.

NOTE
ARP will be generated for a forwarded packet destination IP address when an interface is configured as the next hop.

To configure an IP static route with an IP physical interface as the next hop, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Enter the IP address and prefix-length, or enter the IP address and network mask for the route destination network. On the same command line, enter the keyword **ethernet** followed by the interface number to be used as next hop.

```
device(config)# ip route 10.128.2.69 255.255.255.0 ethernet 1/4
```

   This example configures an IP static route with a destination network address of 10.128.2.69, a network mask of 255.255.255.0, and Ethernet port 1/4 as the next hop.

4. Enter the IP address and prefix-length for the route destination network. On the same command line, enter the appropriate **gigabitethernet** keyword for the line speed followed by the Rbridge ID slot and port of the interface number to be used as next hop.

```
device(config-rbridge-id-30)# ip route 10.128.2.69/24 tengigabitethernet 4/1
```

   This example configures an IP static route with a destination network address of 10.128.2.69 and a prefix-length of 24 on a 10-Gbps connection to the next-hop interface 4/1.

5. Enter the IP address and prefix length for the route destination network. On the same command line, enter the keyword **ethernet** followed by the interface number to be used as next hop.

```
device(config)# ip route 10.128.2.69/24 ethernet 1/4
```

   This example configures an IP static route with a destination network address of 10.128.2.69, a prefix length of /24, and Ethernet port 1/4 as the next hop.

6. Enter the IP address and prefix length for the route destination network. On the same command line, enter the keyword **ethernet** followed by the interface number to be used as next hop.

```
device(config)# ip route 10.128.2.69/24 ethernet 0/4
```

   This example configures an IP static route with a destination network address of 10.128.2.69, a prefix length of /24, and Ethernet port 0/4 as the next hop.

The following example configures an IP static route to destination network addresses beginning with 10.0.0.0 through the next-hop interface 2/1.

```
device# configure terminal
device(config)# ip route 10.0.0.0/24 ethernet 2/1
```

The following example configures an IP static route to destination network addresses beginning with 10.0.0.0 through the next-hop interface 0/1.

```
device# configure terminal
device(config)# ip route 10.0.0.0/24 ethernet 0/1
```

The following example configures an IP static route with a destination network address of 10.128.2.69 and a prefix-length of 24 on a 10-Gbps connection to the next-hop interface 30/4/1.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 10.128.2.69/24 tengigabitethernet 4/1
```

Configuring a virtual interface as next hop

The virtual interface you use for the static route's next hop must have at least one IP address configured on it. The address does not need to be in the same subnet as the destination network.

To configure an IP static route that uses a virtual interface as the next hop, follow these steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the RBridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Enter the IP destination address and the network mask or prefix-length. On the same command line, enter the keyword **ve** followed by the appropriate ID number.

   ```
   device(config)# ip route 10.128.2.0 255.255.255.0 ve 3
   ```

4. Enter the IP destination address and the prefix-length. On the same command line, enter the keyword **ve** followed by the appropriate VLAN number.

   NOTE
   After the keyword **ve**, you can also enter an interface number (slot/port) or 0 to designate a null route.

   ```
   device(config-rbridge-id-30)# ip route 10.128.2.0/24 ve 3
   ```

5. Enter the IP destination address and the prefix-length. On the same command line, enter the keyword **ve** followed by the appropriate VLAN number.

   ```
   device(config)# ip route 10.128.2.0/24 ve 3
   ```

The following example configures an IP static route with a destination address of 10.128.2.0, a prefix-length of /24, and a virtual interface (ve 3) as the next hop.

The following example configures an IP static route with a destination address of 10.128.2.0, a prefix-length of /24, and a virtual interface (ve 3) as the next hop.

```
device# configure terminal
device(config)# ip route 10.128.2.0/24 ve 3
```

```
device# configure terminal
device(config)# ip route 10.128.2.0/24 ve 3
```

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 10.128.2.0/24 ve 3
```

Configuring a static route with a VRF as next hop

A VRF can be designated as the next hop toward a destination via a valid port, IP address, or virtual interface.

The VRF must be an existing VRF, and any physical port referenced in the next hop must have a valid IP address for the route to be added to the routing table.

To configure an IP static route that uses a VRF as the next hop, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Enter the IP destination address and the prefix-length. On the same command line, enter the keyword **next-hop-vrf** followed by the appropriate VRF name. Then specify the next hop as an IP address, physical interface, virtual interface, or null route.

   NOTE
   Using a VRF with a physical port as next hop allows for VRF route leaking for all incoming traffic intended for the destination IP address.

```
device(config)# ip route 10.0.0.0/24 next-hop-vrf red ethernet 1/1
device(config)# ip route 10.0.0.0/24 next-hop-vrf red ethernet 0/1
device(config-rbridge-id-30)# ip route 10.0.0.0/24 next-hop-vrf red port-channel 1
```

   This example routes traffic for IP address 10.0.0.0/24 through VRF red over port 1/1.

   This example routes traffic for IP address 10.0.0.0/24 through VRF red over port 0/1.

   This example routes traffic for IP address 10.0.0.0/24 through VRF red over port channel 1.

```
device(config)# ip route 10.0.0.0/24 next-hop-vrf red 11.1.2.3
device(config-rbridge-id-30)# ip route 10.0.0.0/24 next-hop-vrf red 11.1.2.3
```

   This example routes traffic for IP address 10.0.0.0/24 through VRF red via IP address 11.1.2.3.

```
device(config)# ip route 10.0.0.0/24 next-hop-vrf red ve 3
device(config-rbridge-id-30)# ip route 10.0.0.0/24 next-hop-vrf red ve 3
```

   This example routes traffic for IP address 10.0.0.0/24 through VRF red over virtual interface 3.

```
device(config)# ip route 10.0.0.0/24 next-hop-vrf red null 0
```

   This example discards the traffic for IP address 10.0.0.0/24 through VRF red using a null route.

The following example creates a static route to destinations with 10.0.0.0 IP addresses through VRF red via port 2/1.

The following example creates a static route to destinations with 10.0.0.0 IP addresses through VRF red via port 0/1.

The following example creates a static route to destinations with 10.0.0.0 IP addresses through VRF red via 40-Gbps port 30/2/1.

```
device(config)# ip route 10.0.0.0/24 next-hop-vrf red ethernet 2/1
device(config)# ip route 10.0.0.0/24 next-hop-vrf red ethernet 0/1
device(config-rbridge-id-30)# ip route 10.0.0.0/24 next-hop-vrf red fortygigabitethernet 2/1
```

Configuring an MPLS next hop for an IPv4 static route

You can set the next hop for a static route to the egress router of an LSP tunnel if the destination route is contained in the MPLS routing table.

The destination route must be contained in the MPLS routing table.

MPLS route resolution must be enabled for an LSP route to be active.

In this configuration, the static route is updated with the LSP routes and reverts to its original next-hop interface when this feature is disabled or when the LSP goes down. The route through the LSP can also be used as the default route.

To enable the MPLS next-hop feature for IPv4 static routes and configure routes that use the feature, follow these steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the following command.

   ```
   device(config)# ip route next-hop-enable-mpls
   ```

   This command enables static routes to use an LSP tunnel as next hop.

   NOTE
   Use the **no** form of the command to disable MPLS next hops for static routes.

3. Configure a static route with an LSP next hop.

   ```
   device(config)# ip route 10.12.12.12/32 lsp-next-hop extreme1 distance 110 3
   ```

   This example sets up an LSP tunnel as the next hop using its pre-configured name.

4. To verify the destination route, enter the **show ip route** command.

   ```
   device(config)# exit
   device# show ip route
   Total number of IP routes: 6
   Type Codes - B:BGP D:Connected I:ISIS S:Static R:RIP O:OSPF; Cost - Dist/Metric
           Destination        Gateway         Port           Cost           Type
   1       10.47.6.0/24       DIRECT          mgmt 1         0/0            D
   2       10.11.11.11/32     DIRECT          loopback 1     0/0            D
   3       10.12.12.12/32     10.1.0.1        eth 5/1        110/3          O
           10.12.12.12/32     10.12.12.12     lsp extremel   110/3          O
           10.12.12.12/32     10.12.12.12     lsp t11        110/3          O <----
   4       10.13.13.13/32     10.1.0.1        eth 5/1        110/2          O
   5       10.1.0.0/24        DIRECT          eth 5/1        0/0            D
   6       10.1.0.0/24        10.1.0.1        eth 5/1        110/2          O
   device#
   ```

   As shown in the example, there is a route to the destination network 10.12.12.12/32 through the gateway at IP address 10.12.12.12, which is on LSP 11 (lsp t11). The route has an administrative distance of 110 and a cost metric of 3.

5. To verify that an LSP is up and that MPLS has a route to it, enter the **show mpls lsp** and **show mpls route** commands as shown in the following examples.

```
device# show mpls lsp
Note: LSPs marked with * are taking a Secondary Path
                                Admin  Oper  Tunnel   Up/Dn  Retry   Active
Name           To               State State  Intf     Times  No.     Path
t6             10.12.12.13      UP    DOWN    --      0      292     --
t5             10.12.12.12      UP    UP     tnl3     1      0       --
extremel       10.12.12.12      UP    UP     tnl1     1      0       --     <----
t7             10.12.12.12      UP    UP     tnl5     1      0       one
extremelongl   10.12.12.12      UP    UP     tnl9     1      0
--
t4             10.12.12.12      UP    UP     tnl2     1      0       --
t1             10.12.12.12      UP    UP     tnl0     1      0       --
```

The following example enables MPLS as a next hop. It configures a static route to the destination network addresses 10.12.12.12/32 through the LSP named extreme1. The route has an administrative distance of 110 and a metric cost of 3.

```
device# configure terminal
device(config)# ip route next-hop-enable-mpls
device(config)# ip route 10.12.12.12/32 lsp-next-hop extreme1 distance 110 3
```

Configuring a static route for use with a route map

You can configure a static route with a tag that can be referenced in a route map.

Perform these steps to configure a static route with a tag that can be referenced in a route map.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Enter the **ip route** command followed by the destination network IP address and prefix-length and the next-hop IP address. On the same line, enter the keyword **tag** followed by a decimal tag number.

```
device(config)# ip route 10.0.0.0/24 10.1.1.1 tag 3
```

```
device(config-rbridge-id-30)# ip route 10.0.0.0/24 10.1.1.1 tag 3
```

> NOTE
> An address mask may be used instead of the prefix-length (such as 255.255.255.0 instead of /24).

The following example creates an IP static route to destination IP addresses beginning with 10.0.0.0 through the next-hop address 10.1.1.1. The static route includes the tag "3" for later use in a route map.

```
device# configure terminal
device(config)# ip route 10.0.0.0/24 10.1.1.1 tag 3
```

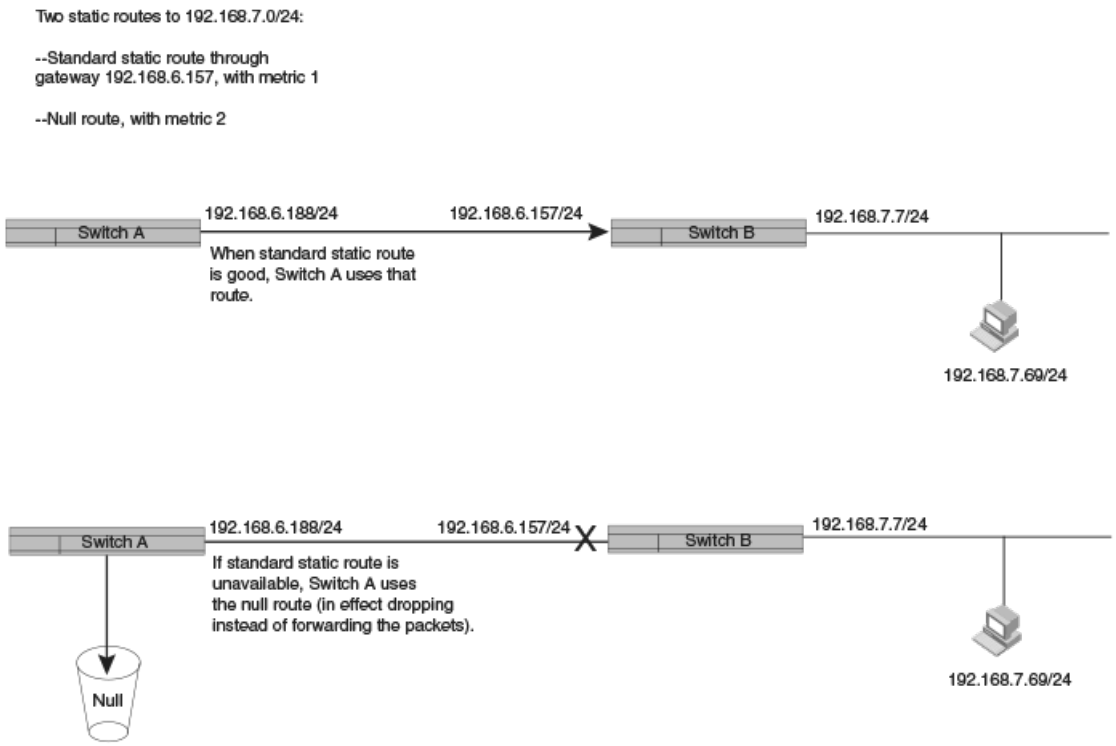
```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 10.0.0.0/24 10.1.1.1 tag 3
```

Configuring a null route

You can configure a null static route to drop packets to a certain destination. This is useful when the traffic should not be forwarded if the preferred route is unavailable.

The following figure depicts how a null static route works with a standard route to the same destination.

FIGURE 5 Null route and standard route to same destination



To configure a null route with a lower priority than the preferred route, perform the following steps.

1. **NOTE**
   You cannot add a null static route to a network if there is already a static route of any type with the same metric you specify for the null route.

   Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the Rbridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Configure the preferred route to a destination.

   ```
   device(config)# ip route 192.168.7.0/24 192.168.6.157
   device(config-rbridge-id-30)# ip route 192.168.7.0/24 192.168.6.157
   ```

   This example creates a static route to destination network addresses that have an IP address beginning with 192.168.7.0. These destinations are routed through the next-hop gateway 192.168.6.157. The route carries the default metric of 1.

4. Configure the null route to the same destination with a higher metric.

   ```
   device(config)# ip route 192.168.7.0/24 null0 2
   ```

   This example creates a null static route to the same destination. The metric is set higher so that the preferred route is used if it is available. When the preferred route becomes unavailable, the null route is used, and traffic to the destination is dropped.

5. Configure a route to the same destination, followed by the keyword **null**, a space, and a zero. On the same line, enter a cost metric that is higher than the metric for the preferred route.

   ```
   device(config)# ip route 192.168.7.0/24 null 0 2
   ```

   This example creates a null static route to the same destination. The metric is set higher so that the preferred route is used if it is available. When the preferred route becomes unavailable, the null route is used, and traffic to the destination is dropped.

6. Configure a route to the same destination, followed by the keyword **null 0**. On the same line, enter a cost metric that is higher than the metric for the preferred route.

   ```
   device(config-rbridge-id-30)# ip route 192.168.7.0/24 null 0 2
   ```

   This example creates a null static route to the same destination. The metric is set to 2, which is higher than the metric for the preferred route. When the preferred route becomes unavailable, the null route is used, and traffic to the destination is dropped.

The following example creates a primary route to all destinations beginning with 192.168.7.0. It creates an alternative null route to drop the packets when the primary route is not available.

```
device# configure terminal
device(config)# ip route 192.168.7.0/24 192.168.6.157
device(config)# ip route 192.168.7.0/24 null0 2

device# configure terminal
device(config)# ip route 192.168.7.0/24 192.168.6.157
device(config)# ip route 192.168.7.0/24 null 0 2

device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 192.168.7.0/24 192.168.6.157
device(config-rbridge-id-30)# ip route 192.168.7.0/24 null 0 2
```

Dropping traffic to the null route in hardware

As an option, you can configure the device to drop packets sent to the null route in hardware.

You can program the CAM discard traffic sent to the null0 interface to improve forwarding efficiency and reduce the burden on the device CPU.

To assign the default route as the null route and drop packets in hardware, follow these steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Configure the default route (0.0.0.0 0.0.0.0) as the null route.

   ```
   device(config)# ip route 0.0.0.0 0.0.0.0 null0
   ```

3. Configure the hardware to drop packets for the default route.

   ```
   device(config)# ip hw-drop-on-def-route
   ```

The following example configures the default IP route as the null route and specifies that packets to the default route be dropped in hardware.

```
device# configure terminal
device(config)# ip route 0.0.0.0 0.0.0.0 null0
device(config)# ip hw-drop-on-def-route
```

Configuring CAM default route aggregation

You can enable the CAM Default Route Aggregation feature to prevent blocking CAM space.

Configuring the device to drop traffic sent to the default IP route address in hardware causes the device to program 32-bit host CAM entries for each destination address using the default route, which could consume the CAM space. To prevent this, enable the CAM Default Route Aggregation feature by entering the following commands.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the following command.

   ```
   device(config)# ip dr-aggregate
   ```

The following example configures the null route as the default route, discards packets to the null route in hardware, and configures the CAM Default Aggregation feature to prevent consumption of CAM space.

```
device# configure terminal
device(config)# ip route 0.0.0.0 0.0.0.0 null0
device(config)# ip hw-drop-on-def-route
device(config)# ip dr-aggregate
```

Configuring a default static route

You can manually create a default static route that the router uses if there are no other default routes to a destination.

If the default route is a protocol route, that protocol needs to be enabled to resolve static routes. Use the **ip route next-hop** command to allow protocol resolution through the default route.

If the default route itself is a static route, you must configure the **ip route next-hop-enable-default** command to resolve other static routes through the default route. You may also configure recursive lookup to resolve the next hop.

Perform these steps to configure a default route.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the Rbridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Enter 0.0.0.0 0.0.0.0 as the destination route and network mask. On the same line, enter a valid next-hop address.

   ```
   device(config)# ip route 0.0.0.0 0.0.0.0 10.24.4.1
   ```

   The example creates a default route through IP address 10.24.4.1.

4. Enter 0.0.0.0/0 as the destination route and prefix-length. On the same line, add the next-hop as an IP address, or enter the key phrase **null 0** to create a null route.

   ```
   device(config)# ip route 0.0.0.0/0 10.24.4.1
   ```

   The example creates a default route through IP address 10.24.4.1.

   ```
   device(config)# ip route 0.0.0.0/0 null 0
   ```

   The example creates a default route that is a null route.

   NOTE
   You cannot create a default route to a virtual or physical interface.

5. Enter 0.0.0.0/0 as the destination route and prefix-length. On the same line, enter a valid next-hop address.

   ```
   device(config-rbridge-id-30)#  ip route 0.0.0.0/0 10.24.4.1
   ```

   The example creates a default route through next-hop address 10.24.4.1.

6. (Optional) Enable the default network route for static route next-hop resolution.

   ```
   device(config)# ip route next-hop-enable-default
   ```

   NOTE
   This command can be independently applied on a per-VRF basis.

7. (Optional) Configure next-hop recursive lookup to resolve the next-hop gateway.

   ```
   device# configure terminal
   device(config)# ip route next-hop-recursion
   ```

The following example configures static routing next-hop recursion to three levels (the default). It configures the network default static route through next-hop IP address 10.24.4.1 and allows the default route to resolve other static routes.

NOTE
You can specify a level of recursion up to 10.

```
device# configure terminal
device(config)# ip route next-hop-recursion
device(config)# ip route 0.0.0.0 0.0.0.0 10.24.4.1
device(config)# ip route next-hop-enable-default
```

The following example configures a network default static route that uses IP address 10.24.4.1 as the next hop.

```
device# configure terminal
device(config)# ip route 0.0.0.0/0 10.24.4.1
```

The following example configures the network default static route through next-hop 10.24.4.1.

```
device# configure terminal
device(config)#
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 0.0.0.0/0 10.24.4.1
```

Resolving a static route using other static routes

You can use next-hop recursive lookup to resolve a static route.

Perform these steps to enable next-hop recursive lookup.

NOTE
This command can be independently applied on a per-VRF basis.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the following command, and, as an option, specify the level of recursion.

```
device(config)# ip route next-hop-recursion
```

This example configures recursive static route lookup to three levels (the default).

The following example configures recursive static route lookup to five levels.

```
device# configure terminal
device(config)# ip route next-hop-recursion 5
```

Resolving the next hop through a protocol

You can use routes from a specific protocol to resolve a configured static route.

Perform these steps to resolve the next hop for a static route using learned routes from a protocol.

NOTE
Connected routes are always used to resolve static routes.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable resolution through the desired protocol.

These protocol options are available:

- bgp
- isis
- ospf
- rip

```
device(config)# ip route next-hop ospf
```

This example enables route resolution through OSPF.

```
device(config)# ip route next-hop bgp
```

This example resolves static routes through BGP. Both iBGP and eBGP routes are used to resolve the routes.

```
device(config)# ip route next-hop isis
```

This example resolves static routes through ISIS.

Static routes between VRFs

You can configure a static route next hop to be in a different VRF.

Static routes can be configured between VRFs in the following ways:

- From the default VRF to a non-default VRF
- From a non-default VRF to a non-default VRF
- From a non-default VRF to the default VRF
- From one VRF to an IP interface in a different VRF.
- Between two named VRFs
- From one VRF to an IP interface in a different VRF.

NOTE
RPF is not supported on static routes between VRFs.

NOTE
For information on disabling gratuitous ARP requests on a VRF IP interface, refer to "Disabling gratuitous ARP requests for local proxy ARP."

Configuring a static route from the default VRF to a non-default VRF

You can create an IPv4 static route from the default VRF to a non-default VRF.

VRFs included in the route configuration must be valid VRFs.

To create an IPv4 static route between the default VRF and a non-default VRF, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter configuration mode for the default VRF.

```
device(config)# vrf default-vrf
```

3. Create a Route Descriptor.

```
device(config-vrf-default-vrf)# rd 2:1
```

4. Specify that IPv4 addressing is to be used on the route.

```
device(config-vrf-default-vrf)# address-family ipv4 unicast
```

5. Configure the destination address and mask, the next hop VRF, and its IP address.

```
device(config-vrf-default-vrf-ipv4u)# ip route 10.128.2.69/24 next-hop-vrf blue 10.1.1.1
```

The following example creates an IPv4 static route on the default VRF to network addresses beginning with 10.128.2.69 (Route Descriptor 2:1). It uses VRF Blue, with an IP address of 10.1.1.1, as the next-hop gateway.

```
device# configure terminal
device(config)# vrf default-vrf
device(config-vrf-default-vrf)# rd 2:1
device(config-vrf-default-vrf)# address-family ipv4 unicast
device(config-vrf-default-vrf-ipv4u)# ip route 10.128.2.69/24 next-hop-vrf blue 10.1.1.1
```

Configuring a static route from a non-default VRF to the default VRF

VRFs included in the configuration must be valid VRFs. Brocade recommends that you create a route descriptor for the destination.

To create an IPv4 static route from a non-default VRF to the default VRF, follow these steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter configuration mode for the non-default VRF (the VRF named "Red" in this example).

   ```
   device(config)# vrf red
   ```

3. Create a Route Descriptor.

   ```
   device(config-vrf-red)# rd 3:1
   ```

4. Specify that IPv4 addressing is to be used on the route.

   ```
   device(config-vrf-red)# address-family ipv4 unicast
   ```

5. Configure the static route destination address. Specify the default VRF and its IP address as the next-hop gateway.

   ```
   device(config-vrf-red-ipv4u)# ip route 10.128.2.69/24 next-hop-vrf default-vrf 10.1.1.1
   ```

The following example configures a static route from VRF Red. It assigns the Route Descriptor 3:1 to the route and specifies that it is an IPv4 route. The example configures the route destination as network addresses that match 10.128.2.69/24. The default VRF, with an IP address of 10.1.1.1, is configured as the next-hop gateway.

```
device# configure terminal
device(config)# vrf red
device(config-vrf-red)# rd 3:1
device(config-vrf-red)# address-family ipv4 unicast
device(config-vrf-red-ipv4u)# ip route 10.128.2.69/24 next-hop-vrf default-vrf 10.1.1.1
```

Configuring a static route between two named VRFs

VRFs included in the route configuration must be valid VRFs.

To create an IPv4 static route between two named VRFs, follow these steps.

To create an IPv4 static route between two non-default VRFs, follow these steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the Rbridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Enter configuration mode for the source VRF (VRF red in this example).

   ```
   device(config)# vrf red
   device(config-rbridge-id-30)# vrf red
   ```

4. Create a Route Descriptor.

```
device(config-vrf-red)# rd 1:1
device(config-rbridge-id-30-vrf-red)# rd 1:1
```

5. Specify that IPv4 addressing is to be used on the route.

```
device(config-vrf-red)# address-family ipv4 unicast
device(config-rbridge-id-30-vrf-red)# address-family ipv4 unicast
```

6. Configure the destination address and mask, the next hop VRF, and its IP address.

```
device(config-vrf-red-ipv4-unicast)# ip route 10.128.2.69/24 next-hop-vrf blue 10.1.1.1
device(config-rbridge-id-30-vrf-red-ipv4-unicast)# ip route 10.128.2.69/24 next-hop-vrf blue 10.1.1.1
```

The following example creates an IPv4 static route for VRF Red to network addresses beginning with 10.128.2.69 (Route Descriptor 1:1). It uses VRF blue, with an IP address of 10.1.1.1, as the next-hop gateway.

```
device# configure terminal
device(config)# vrf red
device(config-vrf-red)# rd 1:1
device(config-vrf-red)# address-family ipv4 unicast
device(config-vrf-red-ipv4-unicast)# ip route 10.128.2.69/24 next-hop-vrf blue 10.1.1.1

device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# vrf red
device(config-rbridge-id-30-vrf-red)# rd 1:1
device(config-rbridge-id-30-vrf-red)# address-family ipv4 unicast
device(config-rbridge-id-30-vrf-red-ipv4-unicast)# ip route 10.128.2.69/24 next-hop-vrf blue 10.1.1.1
```

Configuring an IP static interface route across VRFs

You can configure an IP Static interface route from one VRF to an IP interface in a different VRF.

VRFs configured in static routes must already exist.

By configuring static routes between VRFs, you can connect one VRF to a host that is directly connected to a port in a different VRF. You can do this by configuring a static route to the interface that is directly connected to the device with the IP address you want to reach.

Consider VRF A and VRF B with the following characteristics. Then follow these steps to create an interface route between the two VRFs.

VRF A :

Route Distinguisher = 1:1

Interface: ethernet port 1/1

Interface: ethernet port 1/1/1

IP address: 10.0.0.1/24

VRF B :

Route Distinguisher = 2:2

Interface: ethernet port 1/2

Interface: ethernet port 1/1/2

IP address: 10.0.0.1/24

1. Configure VRF A.
   a) Define a route descriptor for VRF A in VRF configuration mode.

```
device# configure terminal
device(config)# vrf A
device(config-vrf-A)# rd 1:1
```

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# vrf A
device(config-rbridge-id-1-vrf-A)# rd 1:1
```

   b) Configure VRF A to use IPv4 addressing.

```
device(config-vrf-A)# address-family ipv4 unicast
device(config-vrf-A-ipv4-unicast)# exit
device(config-vrf-A)# exit
```

```
device(config-rbridge-id-1-vrf-A)# address-family ipv4 unicast
device(config-rbridge-id-1-vrf-A-ipv4-unicast)# exit
device(config-rbridge-id-1-vrf-A)# exit
```

   c) Configure a device interface (port 1/1 in this case) to forward traffic over VRF A.

```
device(config)# interface ethernet 1/1
device(config-if-e10000-1/1)# vrf forwarding A
```

   d) Configure a device interface (port 1/1/1 in this case) to forward traffic over VRF A.

```
device(config-rbridge-id-1)# interface ethernet 1/1/1
device(config-rbridge-id-1-if-e10000-1/1)# vrf forwarding A
```

   e) Assign an IP address to the interface.

```
device(config-if-e10000-1/1)# ip address 10.0.0.1/24
device(config-if-e10000-1/1)# exit
device(config)#
```

```
device(config-rbridge-id-1-if-e10000-1/1)# ip address 10.0.0.1/24
device(config-rbridge-id-1-if-e10000-1/1)# exit
device(config-rbridge-id-1)#
```

2. Configure VRF B.
   a) Define a route descriptor for VRF B in VRF configuration mode.

```
device(config)# vrf b
device(config-vrf-b)# rd 2:2

device(config-rbridge-id-1)# vrf b
device(config-rbridge-id-1-vrf-b)# rd 2:2
```

   b) Configure VRF B to use IPv4 addressing.

```
device(config-vrf-b)# address-family ipv4 unicast
device(config-vrf-b-ipv4-unicast)# exit
device(config-vrf-b)# exit

device(config-rbridge-id-1-vrf-b)# address-family ipv4 unicast
device(config-rbridge-id-1-vrf-b-ipv4-unicast)# exit
device(config-rbridge-id-1-vrf-b)# exit
```

   c) Configure an interface (port 1/2 in this case) to forward traffic from VRF B.

```
device(config)# interface ethernet 1/2
device(config-if-e10000-1/2)# vrf forwarding B
```

   d) Configure an interface (port 1/1/2 in this case) to forward traffic from VRF B.

```
device(config-rbridge-id-1)# interface ethernet 1/1/2
device(config-rbridge-id-1-if-e10000-1/2)# vrf forwarding B
```

   e) Assign an IP address to the interface.

```
device(config-if-e10000-1/2)# ip address 10.0.0.1/24
device(config-if-e10000-1/2)# exit

device(config-rbridge-id-1-if-e10000-1/2)# ip address 10.0.0.1/24
device(config-rbridge-id-1-if-e10000-1/2)# exit
```

3. Configure an IP static route for VRF A that directs traffic to the port that forwards traffic from VRF B.

```
device(config)# vrf a
device(config-vrf-A)# ip route 10.0.0.1/24 ethernet 1/2
```

   This example creates an IPv4 static route from VRF A to network destinations beginning with 10.0.0.1 via port 1/2, which is the port that forwards traffic for VRF B.

```
device(config-rbridge-id-1)# vrf a
device(config-rbridge-id-1-vrf-A)# ip route 10.0.0.1/24 ethernet 1/1/2
```

   This example creates an IPv4 static route from VRF A to network destinations beginning with 10.0.0.1 via port 1/1/2, which is the port that forwards traffic for VRF B.

The following example configures VRF A to forward traffic over port 1/1 and VRF B to forward traffic over port 1/2. Port 1/2 is configured with an IP address of 10.0.0.1/24. As a last step, the example configures a static route on VRF A to use port 1/2 as the next-hop to network destinations beginning with 10.0.0.1/24, thereby connecting VRF A to VRF B.

```
device# configure terminal
device(config)# vrf A
device(config-vrf-A)# rd 1:1
device(config-vrf-A)# address-family ipv4 unicast
device(config-vrf-A-ipv4-unicast)# exit
device(config-vrf-A)# exit
device(config)# interface ethernet 1/1
device(config-if-e10000-1/1)# vrf forwarding A
device(config-if-e10000-1/1)# ip address 10.0.0.1/24
device(config-if-e10000-1/1)# exit
device(config)# vrf B
device(config-vrf-B)# rd 2:2
device(config-vrf-B)# address-family ipv4 unicast
device(config-vrf-B-ipv4u)# exit
device(config-vrf-B)# exit
device(config)# interface ethernet 1/2
device(config-if-e10000-1/2)# vrf forwarding B
device(config-if-e10000-1/2)# ip address 10.0.0.1/24
device(config-if-e10000-1/2)# exit
device(config)# vrf a
device(config-vrf-A)# ip route 10.0.0.1/24 ethernet 1/2
```

The following example configures VRF A to forward traffic over port 1/1/1 and VRF B to forward traffic over port 1/1/2. Port 1/1/2 is configured with an IP address of 10.0.0.1/24. As a last step, the example configures a static route on VRF A to use port 1/1/2 as the next-hop to network destinations beginning with 10.0.0.1/24, thereby connecting VRF A to VRF B.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# vrf A
device(config-rbridge-id-1-vrf-A)# rd 1:1
device(config-rbridge-id-1-vrf-A)# address-family ipv4 unicast
device(config-rbridge-id-1-vrf-A-ipv4-unicast)# exit
device(config-rbridge-id-1-vrf-A)# exit
device(config-rbridge-id-1)# interface ethernet 1/1/1 <-- Incorrect for NOS
device(config-rbridge-id-1-if-e10000-1/1)# vrf forwarding A
device(config-rbridge-id-1-if-e10000-1/1)# ip address 10.0.0.1/24
device(config-rbridge-id-1-if-e10000-1/1)# exit
device(config-rbridge-id-1)# vrf B
device(config-rbridge-id-1-vrf-B)# rd 2:2
device(config-rbridge-id-1-vrf-B)# address-family ipv4 unicast
device(config-rbridge-id-1-vrf-B-ipv4-unicast)# exit
device(config-rbridge-id-1-vrf-B)# exit
device(config-rbridge-id-1)# interface ethernet 1/1/2 <-- Incorrect for NOS
device(config-rbridge-id-1-if-e10000-1/2)# vrf forwarding B
device(config-rbridge-id-1-if-e10000-1/2)# ip address 10.0.0.1/24
device(config-rbridge-id-1-if-e10000-1/2)# exit
device(config-rbridge-id-1)# vrf a
device(config-rbridge-id-1-vrf-A)# ip route 10.0.0.1/24 ethernet 1/1/2
```

Configuring load sharing and redundancy

You can configure multiple IP static routes to the same destination to set up load sharing or backup routes.

If you configure more than one static route to the same destination with different next-hop gateways but the same metrics, the router load balances among the routes using a basic round-robin method.

If you configure multiple static IP routes to the same destination with different next-hop gateways and different metrics, the router always uses the route with the lowest metric. If this route becomes unavailable, the router fails over to the static route with the next-lowest metric. The following figure depicts two routes with different metrics configured for the same destination.

FIGURE 6 Two static routes to same destination

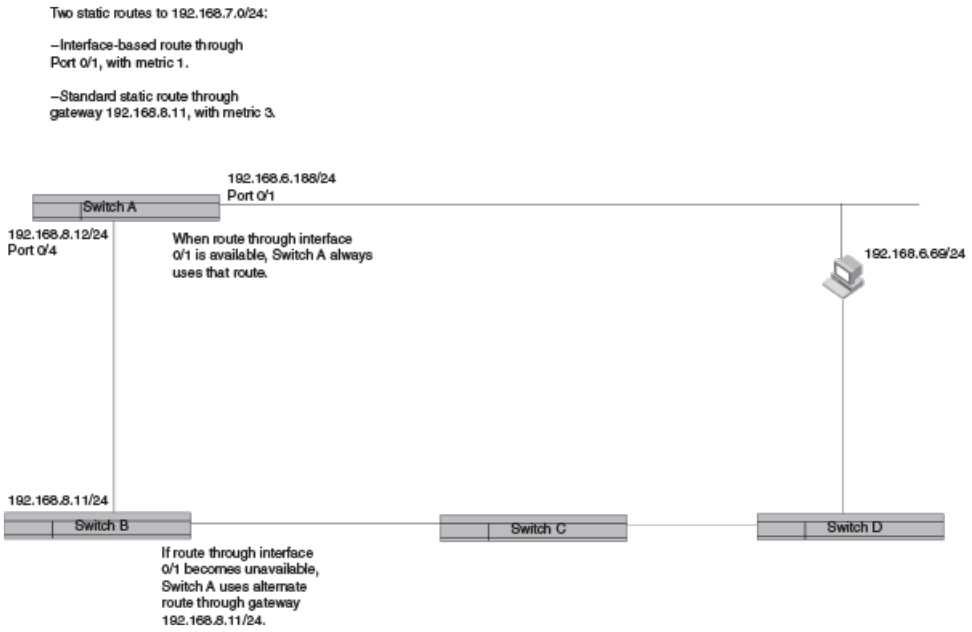


FIGURE 7 Two static routes to same destination

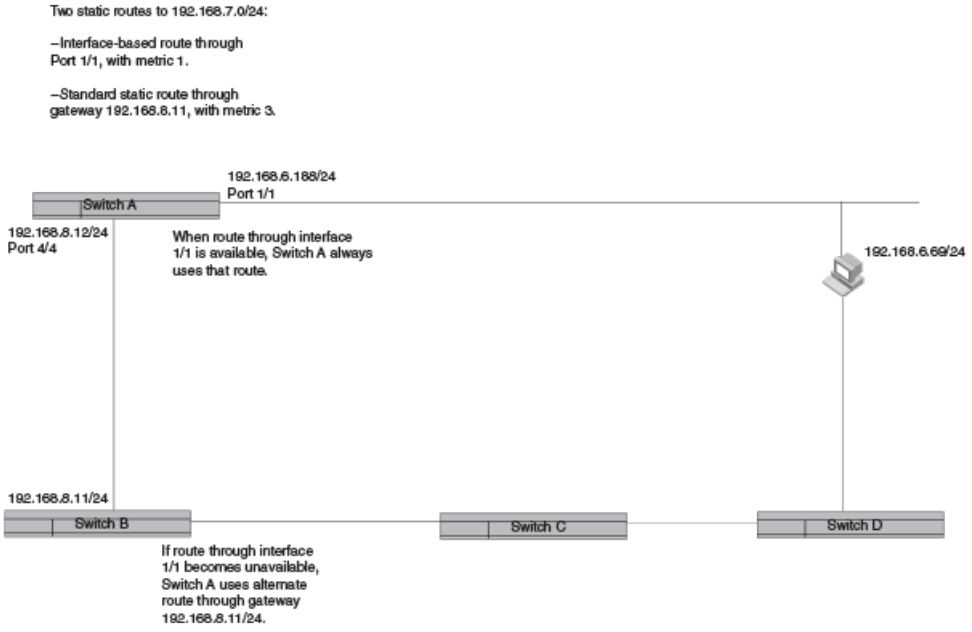
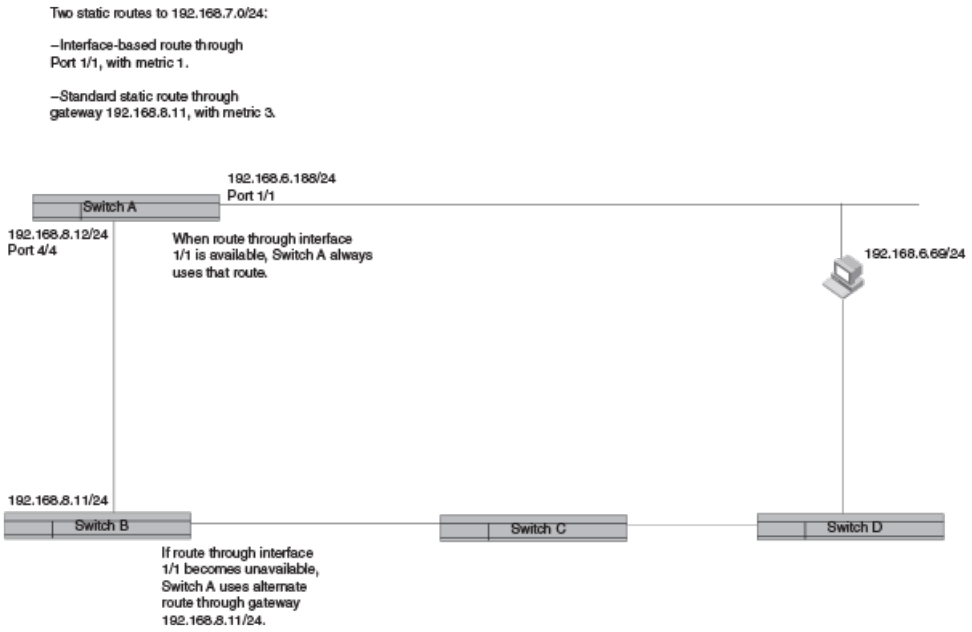


FIGURE 8 Two static routes to same destination



To set up multiple routes for load sharing or redundancy, perform the following steps.

NOTE

You can also use administrative distance to set route priority; however, be sure to give a static route a lower administrative distance than other types of routes, unless you want other route types to be preferred over the static route.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Enter multiple routes to the same destination using different next hops.

```
device(config)# ip route 10.128.2.0/24 10.157.22.1
device(config)# ip route 10.128.2.0/24 10.111.10.1
device(config)# ip route 10.128.2.0/24 10.1.1.1

device(config-rbridge-id-30)# ip route 10.128.2.0/24 10.157.22.1
device(config-rbridge-id-30)# ip route 10.128.2.0/24 10.111.10.1
device(config-rbridge-id-30)# ip route 10.128.2.0/24 10.1.1.1
```

This example creates three next-hop gateways to the destination. Traffic will alternate among the three paths through next-hop 10.157.22.1, next-hop 10.111.10.1, and next hop 10.1.1.1.

4. To prioritize the three routes, use different metrics for each of the three potential next hops.

```
device(config)# ip route 10.128.2.0/24 10.157.22.1
device(config)# ip route 10.128.2.0/24 10.111.10.1 2
device(config)# ip route 10.128.2.0/24 10.1.1.1 3

device(config-rbridge-id-30)# ip route 10.128.2.0/24 10.157.22.1
device(config-rbridge-id-30)# ip route 10.128.2.0/24 10.111.10.1 2
device(config-rbridge-id-30)# ip route 10.128.2.0/24 10.1.1.1 3
```

This example creates three alternate routes to the destination. The primary next hop is 10.157.22.1, which has the default metric of 1 (the default metric is not entered in the CLI). If this path is not available, traffic is directed to 10.111.10.1, which has the next lowest metric of 2. If the second path fails, traffic is directed to 10.1.1.1, which has a metric of 3.

Displaying IPv4 static routes

You can check configured IPv4 routes, static routes, directly connected routes, routes configured for different protocols, the cost associated with each route, and the time the route has been available.

1. To show the IP static routes as configured, enter the **show running-config ip route** command.

   The following example shows default routes, null routes, and routes configured to physical and virtual interfaces.

```
device# show running-config ip route
ip route 0.0.0.0/0 11.1.1.20
ip route 0.0.0.0/0 12.1.1.20
ip route 11.16.1.0/24 11.1.1.20
ip route 11.12.1.0/24 null 0 2 distance 4 tag 10
ip route 11.13.1.0/24 null 0
ip route 11.14.1.0/24 ethernet 3/14
ip route 11.15.1.0/24 ve 10
ip route 11.17.1.0/24 ethernet 3/14 4 distance 5 tag 6
device#
```

   The following example shows default routes, null routes, and routes configured to physical and virtual interfaces.

```
device# show running-config ip route
ip route 0.0.0.0/0 11.1.1.20
ip route 0.0.0.0/0 12.1.1.20
ip route 11.16.1.0/24 11.1.1.20
ip route 11.12.1.0/24 null 0 2 distance 4 tag 10
ip route 11.13.1.0/24 null 0
ip route 11.14.1.0/24 ethernet 0/14
ip route 11.15.1.0/24 ve 10
ip route 11.17.1.0/24 ethernet 0/14 4 distance 5 tag 6
device#
```

2. To display a list of active static routes and their connection times, at the device prompt, enter the **show ip route static** command.

   The following example displays only static routes available on the device, including a default route at the top.

```
device# show ip route static
IP Routing Table for VRF "default-vrf"
Total number of IP routes: 8
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

0.0.0.0/0
    *via 11.1.1.20, Eth 0/14, [1/1], 5m32s, static, tag 0
11.12.1.0/24
    *via DIRECT, Null0, [4/2], 16m57s, static, tag 10
11.13.1.0/24
    *via DIRECT, Null0, [1/1], 16m12s, static, tag 0
11.14.1.0/24
    *via DIRECT, Eth 0/14, [1/1], 11m58s, static, tag 0
11.16.1.0/24
    *via 11.1.1.20, Eth 0/14, [1/1], 9m46s, static, tag 0
11.17.1.0/24
    *via DIRECT, Eth 0/14, [5/4], 9m2s, static, tag 6
device#
```

   The following example displays only static routes available on the device, including a default route at the top.

```
device# show ip route static
IP Routing Table for VRF "default-vrf"
Total number of IP routes: 8
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

0.0.0.0/0
    *via 11.1.1.20, Eth 3/14, [1/1], 5m32s, static, tag 0
11.12.1.0/24
    *via DIRECT, Null0, [4/2], 16m57s, static, tag 10
11.13.1.0/24
    *via DIRECT, Null0, [1/1], 16m12s, static, tag 0
11.14.1.0/24
    *via DIRECT, Eth 3/14, [1/1], 11m58s, static, tag 0
11.16.1.0/24
    *via 11.1.1.20, Eth 3/14, [1/1], 9m46s, static, tag 0
11.17.1.0/24
    *via DIRECT, Eth 3/14, [5/4], 9m2s, static, tag 6
device#
```

3. To narrow the output to information for a particular route, enter the **show ip route static** command followed by the destination IP address without the mask or prefix-length.

   The following example displays information on a static route with IP address 11.18.1.0/24 using VRF t1 and Ethernet port 0/9 as the next hop.

```
device# configure terminal
device(conf)# ip route 11.18.1.0/24 next-hop-vrf t1 ethernet 0/9
device(conf)# exit

device# show ip route 11.18.1.0
IP Routing Table for VRF "default-vrf"
Total number of IP routes: 13
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

11.18.1.0/24
    *via DIRECT, Eth 0/9, [1/1], 0m51s, static+(t1), tag 0   <-- "static+(t1)" indicates
                                                                 route leaking over VRF t1
device#
device# show ip route static vrf t1
Total number of IP routes: 0
device#
device# show run int e 0/9
interface Ethernet 0/9
vrf forwarding t1
ip proxy-arp
ip address 17.1.1.11/24
no shutdown
!
```

4. To show all active IP routes and their connection times, enter the **show ip route** command.

The following example shows six static routes of eight available IP routes on the device, along with the times the routes have been active.

```
device# show ip route
IP Routing Table for VRF "default-vrf"
Total number of IP routes: 8
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

0.0.0.0/0
    *via 11.1.1.20, Eth 3/14, [1/1], 5m19s, static, tag 0
11.1.1.0/24, attached
    *via DIRECT, Eth 3/14, [0/0], 17m46s, direct, tag 0
11.1.1.11/32, attached
    *via DIRECT, Eth 3/14, [0/0], 17m46s, local, tag 0
11.12.1.0/24
    *via DIRECT, Null0, [4/2], 16m44s, static, tag 10
11.13.1.0/24
    *via DIRECT, Null0, [1/1], 15m59s, static, tag 0
11.14.1.0/24
    *via DIRECT, Eth 3/14, [1/1], 11m45s, static, tag 0
11.16.1.0/24
    *via 11.1.1.20, Eth 3/14, [1/1], 9m33s, static, tag 0
11.17.1.0/24
    *via DIRECT, Eth 3/14, [5/4], 8m49s, static, tag 6
device#
```

The following example shows six static routes of eight available IP routes on the device, along with the times the routes have been active.

```
device# show ip route
IP Routing Table for VRF "default-vrf"
Total number of IP routes: 8
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

0.0.0.0/0
    *via 11.1.1.20, Eth 0/14, [1/1], 5m19s, static, tag 0
11.1.1.0/24, attached
    *via DIRECT, Eth 0/14, [0/0], 17m46s, direct, tag 0
11.1.1.11/24, attached
    *via DIRECT, Eth 0/14, [0/0], 17m46s, local, tag 0
11.12.1.0/24
    *via DIRECT, Null0, [4/2], 16m44s, static, tag 10
11.13.1.0/24
    *via DIRECT, Null0, [1/1], 15m59s, static, tag 0
11.14.1.0/24
    *via DIRECT, Eth 0/14, [1/1], 11m45s, static, tag 0
11.16.1.0/24
    *via 11.1.1.20, Eth 0/14, [1/1], 9m33s, static, tag 0
11.17.1.0/24
    *via DIRECT, Eth 0/14, [5/4], 8m49s, static, tag 6
device#
```

The following example...

The following example shows three configured routes. All routes have been active for over six days.

There are two configured IPv4 routes from the management port. The first is the default route, 0.0.0.0/0. This is a static route ("S") that uses the next-hop gateway 10.25.224.1. The default route has an administrative distance of 1 and a cost metric of 1. The other route is a direct route with no additional administrative or cost metric. A third route is a direct route configured as a loopback.

```
device# show ip route
Total number of IP routes: 3
Type Codes - B:BGP D:Connected I:ISIS O:OSPF R:RIP S:Static; Cost - Dist/Metric
BGP  Codes - i:iBGP e:eBGP
ISIS Codes - L1:Level-1 L2:Level-2
OSPF Codes - i:Inter Area 1:External Type 1 2:External Type 2 s:Sham Link
STATIC Codes - d:DHCPv6
        Destination        Gateway         Port           Cost          Type Uptime src-vrf
1       0.0.0.0/0          10.25.224.1     mgmt 1         1/1           S    6d1h   -
2       1.1.1.1/32         DIRECT          loopback 1     0/0           D    6d1h   -
3       10.25.224.0/24     DIRECT          mgmt 1         0/0           D    6d1h   -
```

The following example shows seven IP routes. Three routes are direct routes, and two of those are connected. All other routes are reached via gateway 1.1.1.2 or 1.1.2.2.

```
device# show ip route
 Total number of IP routes: 7
Type Codes - B:BGP D:Connected O:OSPF R:RIP S:Static U: Unnumbered +: Leaked route; Cost - Dist/Metric
BGP  Codes - i:iBGP e:eBGP
OSPF Codes - i:Inter Area 1:External Type 1 2:External Type 2 s:Sham Link
        Destination        Gateway         Port        Cost          Type Uptime
1       1.1.1.0/24         DIRECT          Te 2/1      0/0           D    1m57s
2       1.1.2.0/24         DIRECT          Te 2/2      0/0           D    0m6s
3       100.1.1.0/24       1.1.1.2         Te 2/1      1/1           S    1m32s
4       100.1.2.0/24       1.1.1.2         Te 2/1      1/1           S    1m16s
5       100.1.3.0/24       1.1.1.2         Te 2/1      1/1           S    1m13s
6       100.2.1.0/24       DIRECT          Te 2/1      1/1           S    0m57s
7       100.3.1.0/24       1.1.1.2         Te 2/1      1/1           S    0m5s
        100.3.1.0/24       1.1.2.2         Te 2/2      1/1           S    0m5s
```

IPv6 Static Routing

- Overview of static routing....115
- Configuring a basic IPv6 static route....117
- Naming an IPv6 static route....118
- Removing an IPv6 static route....118
- Configuring an interface as next hop....122
- Configuring a virtual interface as next hop....124
- Configuring a tunnel as next hop....125
- Configuring a VRF as next hop for an IPv6 static route....126
- Adding metrics to an IPv6 static route....127
- Configuring a null route....129
- Configuring a default static route....131
- Resolving the IPv6 static route through a protocol....132
- Configuring load sharing and redundancy....132
- Adding an IPv6 static route tag for use with route-maps....134
- IPv6 multicast static routes....134
- Configuring IPv6 multicast routes in a non-default VRF....135
- Displaying information on IPv6 static routes....136

Overview of static routing

Static routes can be used to specify desired routes, backup routes, or routes of last resort. Static routing can help provide load balancing.

Static routes are manually configured entries in the existing IPv6 routing table. In setting up static routes, you can specify several types of destinations:

- Destination network, using an IP address and network mask or prefix length
- Default network route
- Next hop router
- Next hop tunnel gateway
- Next-hop network protocol type
- Ethernet interface, typically used for directly attached destination networks
- Virtual interface
- Null interface
- Destination network, using an IP address and prefix length
- Default network route
- Next hop router
- VRF for the next hop
- Ethernet interface, typically used for directly attached destination networks
- Virtual interface
- Null interface
- Destination network, using an IP address and prefix length
- Default network route

- VRF for the next hop
- Ethernet interface, typically used for directly attached destination networks
- Virtual interface
- Null interface

You can influence the preference a route is given in the following ways:

- By setting a route metric higher than the default metric
- By giving the route an administrative distance

Static routes can be configured to serve as any of the following:

- Default routes
- Primary routes
- Backup routes
- Null routes for intentionally dropping traffic when the desired connection fails
- Alternative routes to the same destination to help load balance traffic.

Static route states follow port states

IP static routes remain in the IP route table only as long as the port or virtual interface used by the route is available. If the port or virtual routing interface becomes unavailable, the software removes the static route from the IP route table. If the port or virtual routing interface becomes available again later, the software adds the route back to the route table. This feature allows the router to adjust to changes in network topology. The router does not continue trying to use routes on unavailable paths but instead uses routes only when their paths are available.

In the following example, a static route is configured on Switch A.

FIGURE 9 Example of static route

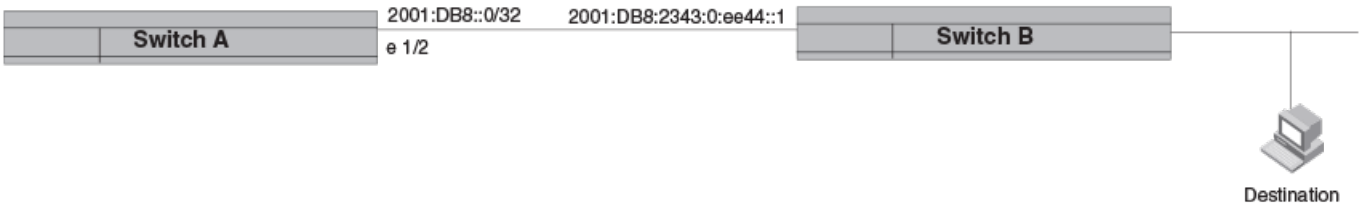
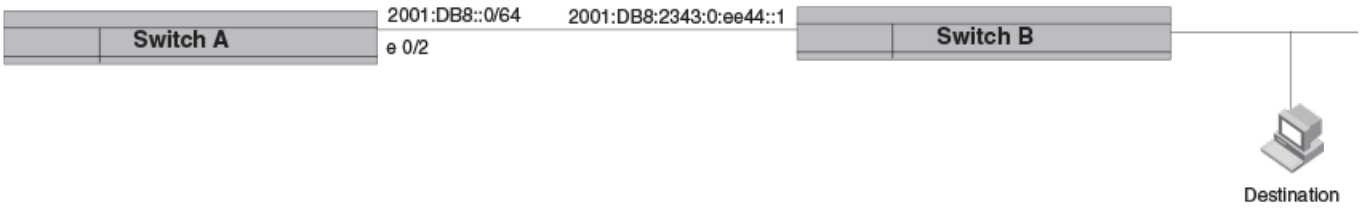


FIGURE 10 Example of static route



The following command configures a static route to 2001:DB8::0/32, using 2001:DB8:2343:0:ee44::1 as the next-hop gateway.

```
device(config)# ipv6 route 2001:DB8::0/32 2001:DB8:2343:0:ee44::1
```

The following command configures a static route to 2001:DB8::0/64, using 2001:DB8:2343:0:ee44::1 as the next-hop gateway.

```
device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1
```

When you configure a static IP route, you specify the destination address for the route and the next-hop gateway or Layer 3 interface through which the Layer 3 device can reach the route. The device adds the route to the IP route table. In this case, Switch A knows that 2001:DB8:2343:0:ee44::1 is reachable through port 1/2, and also assumes that local interfaces within that subnet are on the same port. Switch A deduces that IP interface 2001:DB8::0/32 is also on port 1/2.

When you configure a static IP route, you specify the destination address for the route and the next-hop gateway or Layer 3 interface through which the Layer 3 device can reach the route. The device adds the route to the IP route table. In this case, Switch A knows that 2001:DB8:2343:0:ee44::1 is reachable through port 0/2, and also assumes that local interfaces within that subnet are on the same port. Switch A deduces that IP interface 2001:DB8::0/64 is also on port 0/2.

Configuring a basic IPv6 static route

To configure a basic IPv6 static route, specify the IPv6 destination address, the address mask, and the IPv6 address of the next hop.

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

Before configuring a static IPv6 route, you must enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

To configure a basic IPv6 static route, perform these steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the Rbridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Designate the route destination as an IPv6 address in hexadecimal with 16-bit values between colons, as specified in RFC 2373, and include the address prefix length preceded by a slash. On the same command line, enter the IPv6 address of the next-hop gateway.

   ```
   device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 2001:DB8:0:ee44::1
   ```

4. Designate the route destination as an IPv6 address in hexadecimal with 16-bit values between colons, as specified in RFC 2373, and include the address prefix length preceded by a slash. On the same command line, enter the IPv6 address of the next-hop gateway.

   ```
   device(config)# ipv6 route 2001:DB8::0/32 2001:DB8:0:ee44::1
   ```

5. Designate the route destination as an IPv6 address in hexadecimal with 16-bit values between colons, as specified in RFC 2373, and include the address prefix length preceded by a slash. On the same command line, enter the IPv6 global address or link-local-address of the next-hop gateway.

   ```
   device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:0:ee44::1
   ```

The following example configures an IPv6 static route for a destination network with the prefix 2001:DB8::0/32 and a next-hop gateway with the global address 2001:DB8:0:ee44::1

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 2001:DB8:0:ee44::1
```

The following example configures an IPv6 static route for a destination network with the prefix 2001:DB8::0/32 and a next-hop gateway with the global address 2001:DB8:0:ee44::1

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/32 2001:DB8:0:ee44::1
```

The following example configures an IPv6 static route for a destination network with the prefix 2001:DB8::0/64 and a next-hop gateway with the global address 2001:DB8:0:ee44::1

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:0:ee44::1
```

Naming an IPv6 static route

A non-unique name can be applied to several IPv6 static routes to the same destination to help identify them.

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

An IPv6 static route can be given an optional name. Route names are not unique, and are therefore useful for identifying routes of a particular type, for example, all routes with a particular destination.

Follow these steps to name an IPv6 static route.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Designate the route destination as and IPv6 address in hexadecimal with 16-bit values between colons, as specified in RFC 2373, and include the address prefix length preceded by a slash. On the same command line, enter the IPv6 address of the next-hop gateway. Still on the same command line, add the keyword **name** and the route name as an ASCII string.

   ```
   device(config)# ipv6 route 2001:DB8::0/32 2001:DB8:0:ee44::1 name finance
   ```

The following example configures an IPv6 static route for a destination network with the prefix 2001:DB8::0/32 and a next-hop gateway with the global address 2001:DB8:0:ee44::1. The route has the name "finance." The name may be shared by other routes.

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/32 2001:DB8:0:ee44::1 name finance
```

Removing an IPv6 static route

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

The **no** form of the ipv6 route command must be entered with exact parameters to remove the command. If the route is configured in a non-default VRF, the **no** form of the ipv6 route command must be entered in VRF configuration mode.

Follow these steps to remove an IPv6 static route.

1. (Optional) To view configured routes and confirm exact parameters, enter the command **show ipv6 route** to display the IPv6 route table.

   ```
   device# show ipv6 route
   ```

   This example displays an IPv6 route table with 11 routes in the default VRF.

   ```
   device# show ipv6 route
   IPv6 Routing Table for VRF "default-vrf"
   Total number of IPv6 routes: 11
   '*' denotes best ucast next-hop
   '[x/y]' denotes [preference/metric]

   1200:1201::/64, attached
       *via ::, Eth 2/45, [0/0], 45m29s, direct, tag 0
   1200:1201::1:1/128, attached
       *via ::, Eth 2/45, [0/0], 45m29s, local, tag 0
   1200:1202::/64, attached
       *via ::, Ve 2, [0/0], 45m26s, direct, tag 0
   1200:1202::1:1/128, attached
       *via ::, Ve 2, [0/0], 45m26s, local, tag 0
   2221::/32
       *via 1200:1201::1:2, Eth 2/45, [100/10], 11m41s, static, tag 300
   2222::/48
       *via fe80::205:33ff:fee6:a531, Eth 2/45, [1/1], 43m44s, static, tag 0
   2222::1/128
       *via fe80::205:33ff:fee6:a531, Eth 2/45, [110/1], 0m7s, ospfv3, intra, tag 0
   2223::/64
       *via 1200:1202::1:2, Ve 2, [1/1], 3m45s, static, tag 0
   2224::1/128
       *via fe80::205:33ff:fee6:a501, Ve 2, [1/1], 43m41s, static, tag 0
   fe80::/10, attached
       *via ::, , [0/0], 6h30m, local, tag 0
   ff00::/8, attached
       *via ::, Null0, [0/0], 6h30m, local, tag 0
   ```

   This example displays an IPv6 route table with 11 routes in the default VRF.

   ```
   device# show ipv6 route
   IPv6 Routing Table for VRF "default-vrf"
   Total number of IPv6 routes: 11
   '*' denotes best ucast next-hop
   '[x/y]' denotes [preference/metric]

   1200:1201::/64, attached
       *via ::, Eth 0/40, [0/0], 45m29s, direct, tag 0
   1200:1201::1:1/128, attached
       *via ::, Eth 0/40, [0/0], 45m29s, local, tag 0
   1200:1202::/64, attached
       *via ::, Ve 2, [0/0], 45m26s, direct, tag 0
   1200:1202::1:1/128, attached
       *via ::, Ve 2, [0/0], 45m26s, local, tag 0
   2221::/64
       *via 1200:1201::1:2, Eth 0/40, [100/10], 11m41s, static, tag 300
   2222::/64
       *via fe80::205:33ff:fee6:a531, Eth 0/40, [1/1], 43m44s, static, tag 0
   2222::1/128
       *via fe80::205:33ff:fee6:a531, Eth 0/40, [110/1], 0m7s, ospfv3, intra, tag 0
   2223::/64
       *via 1200:1202::1:2, Ve 2, [1/1], 3m45s, static, tag 0
   2224::1/128
       *via fe80::205:33ff:fee6:a501, Ve 2, [1/1], 43m41s, static, tag 0
   fe80::/10, attached
       *via ::, , [0/0], 6h30m, local, tag 0
   ff00::/8, attached
       *via ::, Null0, [0/0], 6h30m, local, tag 0
   ```

2. (Optional) Enter the **show ipv6 static route** command to narrow the output to static routes only.

```
device# show ipv6 static route
```

3. (Optional) Enter the **show ipv6 route static** command to narrow the output to static routes only.

```
device# show ipv6 route static
IPv6 Configured Static Routes for VRF "default-vrf"

3002:7::/64-> 1200:3::1:2 preference: 1
   nh_vrf (default-vrf)

3002:9::/64-> 1200:4::1:2 preference: 1
   nh_vrf (default-vrf)
```

4. Enter global configuration mode.

```
device# configure terminal
```

5. Enter the RBridge ID.

```
device(config)# rbridge-id 30
```

6. Enter **no** followed by the **ipv6 route** command, including destination and next-hop, as shown in the following example. (You do not need to include cost metric, distance, or tag parameters.)

```
device(config-rbridge-id-30)# no ipv6 route 2224::1/128 fe80::205:33ff:fee6:a501 ve 2


device(config)# no ipv6 route 2224::1/128 fe80::205:33ff:fee6:a501 ve 2
```

This example removes the IPv6 route specified.

The following example removes an existing IPv6 static route from the VRF named corporate.

```
device# configure terminal
device(config)# vrf corporate
device(config-vrf-corporate)# rd 20:10
device(config-vrf-corporate)# address-family ipv6 unicast
device(config-vrf-corporate-ipv6)# no ipv6 route 2002::/64 ethernet 1/1
```

Removing an IPv6 static route in a non-default VRF

Follow these steps to remove an IPv6 static route from a non-default VRF.

1. (Optional) Enter the **show ipv6 route** command to display route information for the non-default vrf.

```
device# show ipv6 route 2225::/32 vrf blue
IPv6 Routing Table for VRF "blue"
Total number of IPv6 routes: 10
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

2225::/32
    *via 1200:3::1:2, Eth 2/47, [1/1], 0m5s, static, tag 0

device# show ipv6 route 2225::/32 vrf blue
Total number of IP routes: 0
device#

device# show ipv6 route 2225::/64 vrf blue
IPv6 Routing Table for VRF "blue"
Total number of IPv6 routes: 10
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

2225::/64
    *via 1200:3::1:2, Eth 0/40, [1/1], 0m5s, static, tag 0

device# show ipv6 route 2225::/64 vrf blue
Total number of IP routes: 0
device#
```

2. Enter VRF configuration mode.

```
device# configure terminal
device(config)# vrf blue
device(config-vrf-blue)# address-family ipv6 unicast
device(vrf-blue-ipv6-unicast)#
```

3. Enter **no** followed by the **ipv6 route** command, including destination and next-hop, as shown in the following example. (You do not need to include cost metric, distance, or tag parameters.)

```
device(vrf-blue-ipv6-unicast)# no ipv6 route 2225::/32 1200:3::1:2

device(vrf-blue-ipv6-unicast)# no ipv6 route 2225::/64 1200:3::1:2
```

4. (Optional) Enter the **show ipv6 route** command for the non-default VRF again to confirm the route has been removed.

```
device# show ipv6 route 2225::/32 vrf blue
Total number of IP routes: 0
device#

device# show ipv6 route 2225::/64 vrf blue
Total number of IP routes: 0
device#
```

The following example verifies a route to be removed, enters VRF configuration mode for the non-default VRF "Blue" to remove the IPv6 route, and confirms that the route has been removed.

```
device# show ipv6 route 2225::/32 vrf blue
IPv6 Routing Table for VRF "blue"
Total number of IPv6 routes: 10
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

2225::/32
    *via 1200:3::1:2, Eth 2/47, [1/1], 0m5s, static, tag 0

device# configure terminal
device(config)# vrf blue
device(config-vrf-blue)# address-family ipv6 unicast
device(vrf-blue-ipv6-unicast)# no ipv6 route 2225::/32 1200:3::1:2
device(vrf-blue-ipv6-unicast)# end

device# show ipv6 route 2225::/32 vrf blue
Total number of IP routes: 0
device#

device# show ipv6 route 2225::/64 vrf blue
IPv6 Routing Table for VRF "blue"
Total number of IPv6 routes: 10
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

2225::/64
    *via 1200:3::1:2, Eth 0/40, [1/1], 0m5s, static, tag 0

device# configure terminal
device(config)# vrf blue
device(config-vrf-blue)# address-family ipv6 unicast
device(vrf-blue-ipv6-unicast)# no ipv6 route 2225::/64 1200:3::1:2
device(vrf-blue-ipv6-unicast)# end

device# show ipv6 route 2225::/64 vrf blue
Total number of IP routes: 0
device#
```

Configuring an interface as next hop

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

Before configuring a static IPv6 route, you must enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

To configure an IPv6 static route with an interface as the next hop as depicted in the following illustration, perform these steps.

FIGURE 11 IPv6 static route with an interface as next hop

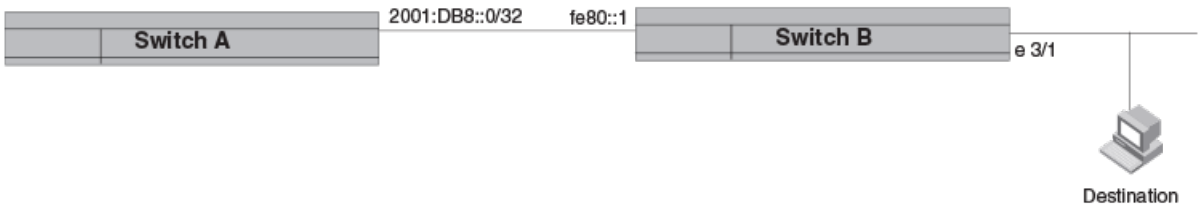
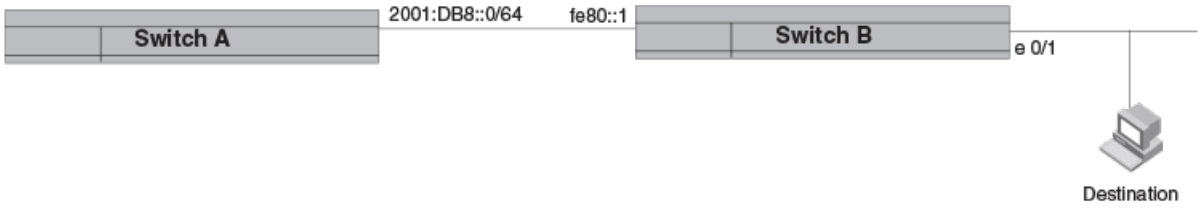


FIGURE 12 IPv6 static route with an interface as next hop



1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Designate the route destination, followed by the next-hop link-local IPv6 address. On the same command line, enter the correct keyword for the line speed (**gigabitethernet**, **tengigabitethernet**, **fortygigabitethernet**, or **hundredgigabitethernet**) followed by the interface (port/slot) to serve as the next-hop.

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 fe80::1 tengigabitethernet 3/1
```

4. Designate the route destination. On the same command line, enter the keyword **ethernet** followed by the interface number as the next-hop, followed by its link-local IPv6 address.

```
device(config)# ipv6 route 2001:DB8::0/32 ethernet 3/1 fe80::1
```

5. Designate the route destination. On the same command line, enter the link-local IPv6 address followed by the keyword **ethernet** and the interface number for the next hop.

```
device(config)# ipv6 route 2001:DB8::0/32 fe80::1 ethernet 3/1
```

6. Designate the route destination. On the same command line, enter the link-local IPv6 address followed by the keyword **ethernet** and the interface number for the next hop.

```
device(config)# ipv6 route 2001:DB8::0/64 fe80::1 ethernet 0/1
```

The following example configures a static IPv6 route for a destination network with the prefix 2001:DB8::0/32 and a next-hop gateway with the link-local address fe80::1 that the Layer 3 switch can access through 10G interface 3/1.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 fe80::1 tengigabitethernet 3/1
```

The following example configures a static IPv6 route for a destination network with the prefix 2001:DB8::0/32 and a next-hop gateway with the link-local address fe80::1 that the Layer 3 switch can access through Ethernet interface 3/1.

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/32 ethernet 3/1 fe80::1
```

The following example configures a static IPv6 route for a destination network with the prefix 2001:DB8::0/32 and a next-hop gateway with the link-local address fe80::1 that the Layer 3 switch can access through Ethernet interface 3/1.

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/32 fe80::1 ethernet 3/1
```

The following example configures a static IPv6 route for a destination network with the prefix 2001:DB8::0/64 and a next-hop gateway with the link-local address fe80::1 that the Layer 3 switch can access through Ethernet interface 0/1.

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/64 fe80::1 ethernet 0/1
```

Configuring a virtual interface as next hop

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

Before configuring a static IPv6 route, you must enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

To configure a basic IPv6 static route with a virtual interface as a next hop, perform these steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the Rbridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Enter the IP address prefix and prefix length for the route destination network.

   ```
   device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32
   ```

   ```
   device(config)# ipv6 route 2001:DB8::0/32
   ```

   ```
   device(config)# ipv6 route 2001:DB8::0/64
   ```

   This example shows the first half of the command, the route destination, IPv6 2001:DB8::0/32 network addresses.

   This example shows the first half of the command, the route destination, IPv6 2001:DB8::0/32 network addresses.

   This example shows the first half of the command, the route destination, IPv6 2001:DB8::0/64 network addresses.

4. On the same command line, add the keyword **ve** followed by the virtual interface ID to be used as the next hop.

   ```
   device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 ve 3
   ```

5. On the same command line, add the keyword **ve** followed by the virtual interface ID to be used as the next hop, along with its link-local address.

   ```
   device(config)# ipv6 route 2001:DB8::0/32 ve 3 fe80::1
   ```

   This example shows the next-hop destination as virtual interface (ve) 3, with a link-local address of fe80::1.

6. On the same command line, enter the link-local address for the destination, followed by the keyword **ve** and the virtual interface ID to be used as the next hop.

   ```
   device# configure terminal
   device(config)# ipv6 route 2001:DB8::0/32 fe80::1 ve 3
   ```

   This example shows the next-hop destination as virtual interface (ve) 3, with a link-local address of fe80::1.

7. On the same command line, enter the link-local address for the destination, followed by the keyword **ve** and the virtual interface ID to be used as the next hop.

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/64 fe80::1 ve 3
```

This example shows the next-hop destination as virtual interface (ve) 3, with a link-local address of fe80::1.

The following example configures an IPv6 static route to IPv6 2001:DB8::0/32 destinations through next-hop virtual interface 3.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 ve 3
```

The following example configures an IPv6 static route to IPv6 2001:DB8::0/32 destinations through next-hop virtual interface 3.

NOTE
The packet destination will be used as next hop over the virtual interface if the next-hop address is not configured in the command.

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/32 ve 3 fe80::1
```

The following example configures an IPv6 static route to IPv6 2001:DB8::0/32 destinations through next-hop virtual interface 3.

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/32 fe80::1 ve 3
```

The following example configures an IPv6 static route to IPv6 2001:DB8::0/64 destinations through next-hop virtual interface 3.

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/64 fe80::1 ve 3
```

Configuring a tunnel as next hop

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

To configure a basic IPv6 static route through a next-hop tunnel, perform these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the IPv6 destination address and prefix, followed by the keyword **ipv6_tnl** and the tunnel ID, followed by the link-local address of the next hop.

```
device(config)# ipv6 route 2001:DB8::0/32 ipv6_tnl 1 fe80::1
```

This example configures Tunnel 1 as the next-hop gateway for IPV6 2001:DB8::0/32 destinations through the link-local address as next hop.

NOTE
To configure a tunnel that allows both IPv4 and IPv6 packets to be carried as IPv4 packets, use the keyword **6to4_tnl** plus the tunnel ID and link-local address to be used as the next-hop gateway.

The following example configures an IPv6 static route to 2001:DB8::0/32 destinations with a next-hop gateway through Tunnel 1.

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/32 ipv6_tnl 1 fe80::1
```

Configuring a VRF as next hop for an IPv6 static route

A non-default VRF can be configured as the next-hop gateway for an IPv6 static route.

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

Before configuring a static IPv6 route, you must enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

> NOTE
> The VRF designated in the procedure must be a valid VRF.

To configure a VRF as the next hop for an IPv6 static route, follow these steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the Rbridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Enter the **ipv6 route** command followed by the IPv6 destination address and prefix length. On the same command line, enter the keyword **next-hop-vrf** followed by the name of the VRF that contains the next-hop gateway router and its IPv6 address.

   ```
   device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 next-hop-vrf partners 2001:DB8:0:ee44::1
   ```

   This example creates an IPv6 static route to IPv6 2001:DB8::0/32 destinations through the VRF named "partners" and the next-hop router with the IPv6 address 2001:DB8:0:ee44::1.

4. Enter the **ipv6 route** command followed by the IPv6 destination address and prefix length. On the same command line, enter the keyword **next-hop-vrf** followed by the name of the VRF that contains the next-hop gateway router and its IPv6 address.

   ```
   device(config)# ipv6 route 2001:DB8::0/32 next-hop-vrf partners 2001:DB8:0:ee44::1
   ```

   This example creates an IPv6 static route to IPv6 2001:DB8::0/32 destinations through the VRF named "partners" and the next-hop router with the IPv6 address 2001:DB8:0:ee44::1.

5. Enter the **ipv6 route** command followed by the IPv6 destination address and prefix length. On the same command line, enter the keyword **next-hop-vrf** followed by the name of the VRF that contains the next-hop gateway router and its IPv6 address.

   ```
   device(config)# ipv6 route 2001:DB8::0/32 next-hop-vrf partners 2001:DB8:0:ee44::1
   ```

   This example creates an IPv6 static route to IPv6 2001:DB8::0/32 destinations through the VRF named "partners" and the next-hop router with the IPv6 address 2001:DB8:0:ee44::1.

6. Enter the **ipv6 route** command followed by the IPv6 destination address and prefix length. On the same command line, enter the keyword **next-hop-vrf** followed by the name of the VRF that contains the next-hop gateway router and its IPv6 address.

   ```
   device(config)# ipv6 route 2001:DB8::0/64 next-hop-vrf partners 2001:DB8:0:ee44::1
   ```

   This example creates an IPv6 static route to IPv6 2001:DB8::0/64 destinations through the VRF named "partners" and the next-hop router with the IPv6 address 2001:DB8:0:ee44::1.

Adding metrics to an IPv6 static route

You can influence how likely a static route is to be used by modifying the cost metric or the administrative distance.

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

Before configuring a static IPv6 route, you must enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

Follow these steps to create an IPv6 static route with cost metrics.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Designate the route destination and next hop, and add a route priority parameter.

Option	Description
Cost metric	The value is compared to the metric for other static routes in the IPv6 route table to the same destination. Two or more routes to the same destination with the same metric load share traffic to the destination. The value may range from 1 through 16. The default is 1. A route with a cost of 16 is considered unreachable.
Administrative distance	This value is compared to the administrative distance of all routes to the same destination. Static routes by default take precedence over learned protocol routes. However, to give a static route a lower priority than a dynamic route, give the static route the higher administrative distance. The value is preceded by the keyword **distance** and can range from 1 to 255. The default is 1. A value of 255 is considered unreachable.

Option	Description
Cost metric	The value is compared to the metric for other static routes in the IPv6 route table to the same destination. Two or more routes to the same destination with the same metric load share traffic to the destination. The value may range from 1 through 16. The default is 1. A route with a cost of 16 is considered unreachable.
Administrative distance	This value is compared to the administrative distance of all routes to the same destination. Static routes by default take precedence over learned protocol routes. However, to give a static route a lower priority than a dynamic route, give the static route the higher administrative distance. The value is preceded by the keyword **distance** and can range from 1 to 254. The default is 1. A value of 255 is considered unreachable.

```
device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1 2
```

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1 2
```

This example configures a static IPv6 route for a destination network with the prefix 2001:DB8::0/64 and a next-hop gateway with the global address 2001:DB8:0:ee44::1 and assigns the route a metric of 2.

```
device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1 distance 254
```

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1 distance 254
```

This example configures a static route with an administrative distance of 254.

The following example configures an IPv6 static route for a destination network with the prefix 2001:DB8::0/64 and a next-hop gateway with the global address 2001:DB8:0:ee44::1. The static route is assigned an administrative distance of 3.

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:0:ee44::1 distance 3
```

```
device# configure terminal
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:0:ee44::1 distance 3
```

Configuring a null route

You can configure a null static route to drop packets to a certain destination. This is useful when the traffic should not be forwarded if the preferred route is unavailable.

> NOTE
> You cannot add a null or interface-based static route to a network if there is already a static route of any type with the same metric you specify for the null or interface-based route.

The following figure depicts how a null static route works with a standard route to the same destination.

FIGURE 13 Null route and standard route to same destination

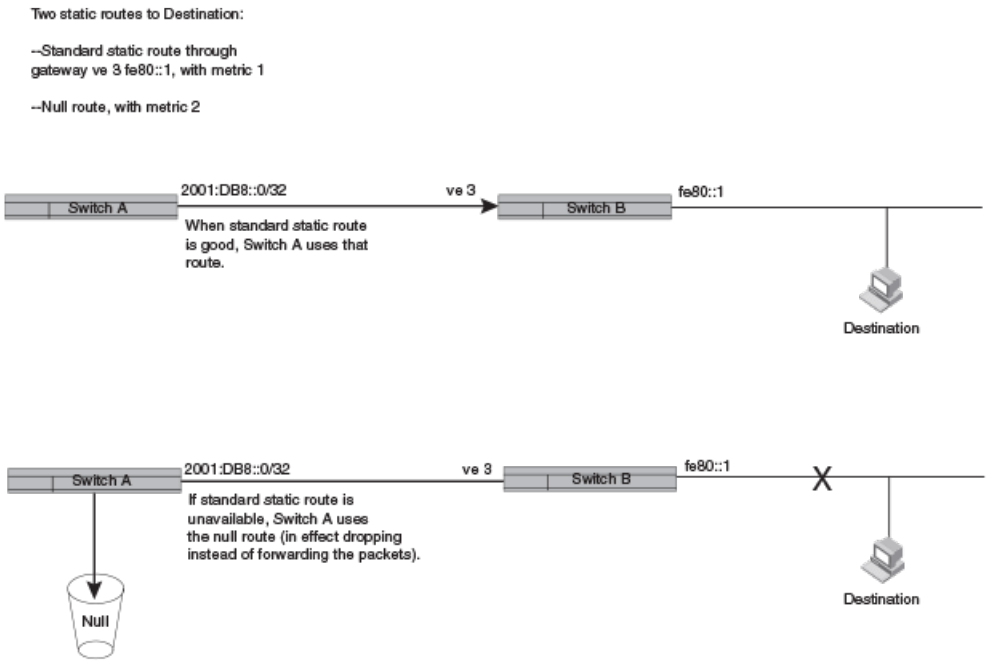
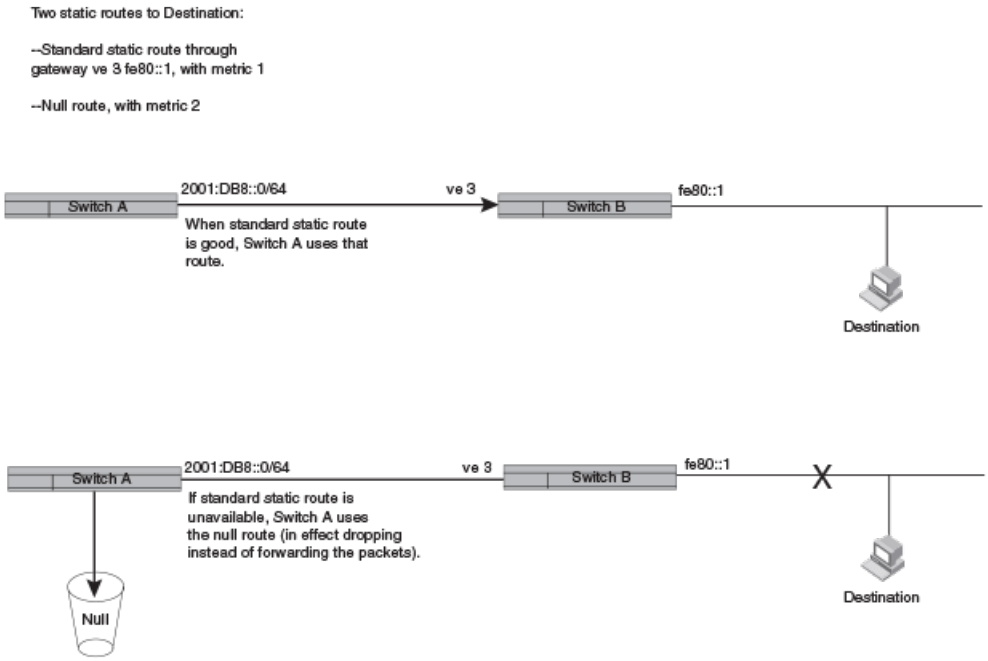


FIGURE 14 Null route and standard route to same destination



The following procedure creates a preferred route and a null route to the same destination. The null route drops packets when the preferred route is not available.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Configure the preferred route to a destination.

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 ve 3
device(config)# ipv6 route 2001:DB8::0/64 ve 3 fe80::1
device(config)# ipv6 route 2001:DB8::0/64 fe80::1 ve 3
```

   This example creates a static route to IPv6 2001 : DB8 : : 0/64 destination addresses. These destinations are routed through link-local address fe80::1 and the next hop gateway virtual interface (ve) 3. The route uses the default cost metric of 1.

4. Configure the null route to the same destination with a higher metric.

```
device(config)# ipv6 route 2001:DB8::0/64 null0 2
device(config)# ipv6 route 2001:DB8::0/64 null 0 2
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 null 0 2
```

   This example creates a null static route to the same destination. The metric is set higher so that the preferred route is used if it is available.

The following example creates a primary route to all 2001 : DB8 : : 0/64 destinations through virtual interface (ve) 3. It creates an alternative null route to drop the packets when the primary route is not available.

```
device# configure terminal
device(config)# ipv6 route 2001 : DB8 : : 0/64 ve 3 fe80::1
device(config)# ipv6 route 2001 : DB8 : : 0/64 null0 2

device# configure terminal
device(config)# ipv6 route 2001 : DB8 : : 0/64 fe80::1 ve 3
device(config)# ipv6 route 2001 : DB8 : : 0/64 null 0 2

device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ipv6 route 2001 : DB8 : : 0/64 ve 3
device(config-rbridge-id-30)# ipv6 route 2001 : DB8 : : 0/64 null 0 2
```

Configuring a default static route

You can manually create a default static route that the router uses if there are no other default routes to a destination.

Because the default route is a static route, you must configure the **ip route next-hop-enable-default** command to resolve other static routes through the default route.

Perform these steps to configure a default route.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the Rbridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Enable the default network route for static route resolution of routes to a particular destination.

   ```
   device(config)# ipv6 route next-hop-enable-default
   ```

   This example enables the default static route to resolve the next hop for IPv6 static routes.

   NOTE
   This command can be independently applied on a per-VRF basis.

4. (Optional) Configure the default route for recursive lookup of the next-hop.

   ```
   device# configure terminal
   device(config)# ipv6 route next-hop-recursion
   ```

   This example allows three levels of recursion in looking up the next hop for any IPv6 static route. The default is 3. You may enter any value from 1 to 10.

5. Enter the following destination route and network mask followed by a valid next-hop address.

   ```
   device(config-rbridge-id-30)#  ipv6 route ::/0 2001:DB8:0:ee44::1

   device(config)# ipv6 route ::/0 2001:DB8:0:ee44::1
   ```

The following example configures a default static route to global IPv6 address 2001:DB8:0:ee44::1. The route is able to resolve static routes using next-hop recursion to three levels (the default).

```
device# configure terminal
device(config)# ipv6 route next-hop-enable-default
device(config)# ipv6 route next-hop-recursion
device(config)# ipv6 route ::/0 2001:DB8:0:ee44::1
```

The following example configures a default static route to global IPv6 address 2001:DB8:0:ee44::1.

```
device# configure terminal
device(config)# ipv6 route ::/0 2001:DB8:0:ee44::1
```

The following example configures a default static route to global IPv6 address 2001:DB8:0:ee44::1.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ipv6 route ::/0 2001:DB8:0:ee44::1
```

Resolving the IPv6 static route through a protocol

You can use routes from another protocol to resolve a static route.

Perform these steps to resolve the next hop for an IPv6 static route using learned routes from BGP, OSPF, RIP, or ISIS protocol.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Designate next-hop resolution through BGP, RIP, OSPF, or ISIS protocol.

The following example specifies that IPv6 static routes can be resolved through directly connected OSPF routers (instead of link-local IPv6 route tables, for example).

```
device# configure terminal
device(config)# ipv6 route next-hop ospf
```

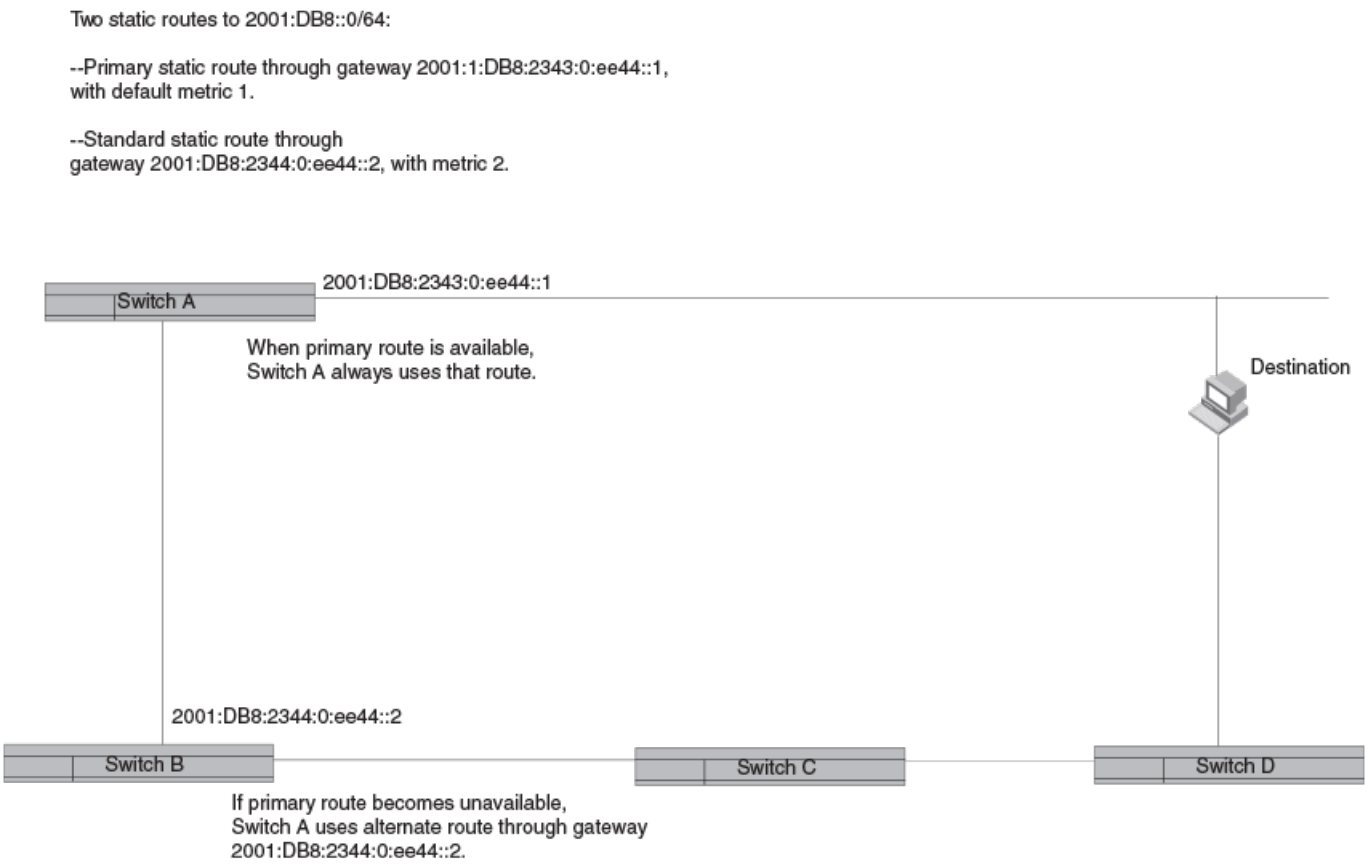
Configuring load sharing and redundancy

You can configure multiple IP static routes to the same destination to set up load sharing or backup routes.

If you configure more than one static route to the same destination with different next-hop gateways but the same metrics, the router load balances among the routes using a basic round-robin method.

If you configure multiple static IP routes to the same destination with different next-hop gateways and different metrics, the router always uses the route with the lowest metric. If this route becomes unavailable, the router fails over to the static route with the next-lowest metric. The following figure depicts multiple routes with different metrics configured for the same destination.

FIGURE 15 Two static routes to same destination



To set up multiple routes for load sharing or redundancy, perform the following steps.

NOTE
You can also use administrative distance to set route priority; however, be sure to give a static route a lower administrative distance than other types of routes, unless you want other route types to be preferred over the static route.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Enter multiple routes to the same destination using different next hops.

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:2344:0:ee44::2

device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1
device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:2344:0:ee44::2
```

This example creates two next-hop gateways for all 2001:DB8::0/64 destinations. Traffic will alternate between the two paths.

4. To prioritize multiple routes, use different metrics for each possible next hop.

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:2344:0:ee44::2 2

device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1
device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:2344:0:ee44::2 2
```

This example creates an alternate route to all 2001:DB8::0/64 destinations. The primary route uses 2001:DB8:2343:0:ee44::1 as the next hop. The route has the default metric of 1. If this path is not available, traffic is directed through 2001:DB8:2344:0:ee44::2, which has the next lowest metric (2).

Adding an IPv6 static route tag for use with route-maps

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

Before configuring a static IPv6 route, you must enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

To configure an IPv6 static route with a tag that can be referenced in a route-map, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Configure the IPv6 static route destination address and next-hop address. On the same command line, enter the keyword tag, followed by the decimal number to be referenced later in a route-map.

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:0:ee44::1 tag 3

device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:0:ee44::1 tag 3
```

The following example configures an IPv6 route to IPv6 2001:DB8::0/64 destinations through next-hop 2001:DB8:0:ee44::1. The route has the tag ID "3," which can be referenced later in a route-map.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:0:ee44::1 tag 3

device# configure terminal
device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:0:ee44::1 tag 3
```

IPv6 multicast static routes

IPv6 multicast routes allow you to control the network path used by multicast traffic.

Static multicast routes are especially useful when the unicast and multicast topologies of a network are different. You can avoid the need to make the topologies similar by instead configuring static multicast routes.

You can configure more than one static IPv6 multicast route. The device by default uses the most specific route that matches a multicast source address. Thus, if you want to configure a multicast static route for a specific multicast source and also configure another multicast static route for all other sources, you can configure two static routes.

You can also influence route preference using cost metrics and administrative distance parameters.

> **NOTE**
> Regardless of the administrative distances, the device always prefers directly connected routes over other routes.

The following example configures an IPv6 multicast static route for a destination network with the prefix 2001:db8::0/32, a next-hop gateway with the global address 2001:db8:0:ee44::1, and an administrative distance of 110.

```
device# configure terminal
device(config)# ipv6 mroute 2001:db8::0/32 2001:db8:0:ee44::1 distance 110
```

Configuring IPv6 multicast routes in a non-default VRF

You can specify a default or non-default VRF for an IPv6 multicast static route. If no VRF is specified, the default VRF is used.

The VRF specified must be an existing VRF.

Follow these steps to create an IPv6 multicast static route in a non-default VRF.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Designate the non-default VRF for the IPv6 multicast static route.

   ```
   device# configure terminal
   device(config)# vrf corporate
   ```

   This example configures "corporate" as the non-default VRF.

3. Give the route a Route Descriptor.

   ```
   device(config-vrf-corporate)# rd 20:10
   ```

   This example configures 20:10 as the route descriptor.

4. Specify the address family as IPv6.

   ```
   device(config-vrf-corporate)# address-family ipv6
   ```

5. Configure the IPv6 static route, including destination IP address, mask prefix, and next-hop information.

   ```
   device(config-vrf-corporate-ipv6)# ipv6 mroute 2002::/64 ethernet 1/1
   ```

   This example configures an IPv6 static route to IP address 2002::/64 destinations via next-hop interface 1/1.

The following example creates an IPv6 multicast route with the RD 20:10 to 2002::/64 IP address via next-hop Ethernet interface 1/1. The route has a cost metric of 5.

```
device# configure terminal
device(config)# vrf corporate
device(config-vrf-corporate)# rd 20:10
device(config-vrf-corporate)# address-family ipv6
device(config-vrf-corporate-ipv6)# ipv6 mroute 2002::/64 ethernet 1/1 fe80::1 5
```

Displaying information on IPv6 static routes

You can consult the IPv6 route table for information on connected, static, and protocol routes.

To display information on IPv6 static routes, use the following commands.

1. To check whether IPv6 is enabled, enter the **show ipv6** command. The command can be entered at the device prompt or in global or interface configuration mode.

```
device# show ipv6
Global Settings
  IPv6 Router-Id: 1.1.1.1   load-sharing path: 4
  unicast-routing enabled, ipv6  allowed to run, hop-limit 64
  reverse-path-check disabled
  nd6 proxy disabled
  host drop cam limit disabled
  urpf-exclude-default disabled
  session-logging-age 5
  selective-routes-download enabled
  No Inbound Access List Set
  No Outbound Access List Set
  source-route disabled, forward-source-route disabled, icmp-redirect disabled  icmp-mpls-response
enabled
OSPF (default VRF): enabled
```

   This example shows IPv6 is enabled on the device.

2. To display IPv6 route table information, enter the **show ipv6 route** command.

```
device# show ipv6 route
IPv6 Routing Table - 2 entries:
Type Codes - B:BGP C:Connected I:ISIS L:Local O:OSPF R:RIP S:Static
BGP  Codes - i:iBGP e:eBGP
ISIS Codes - L1:Level-1 L2:Level-2
OSPF Codes - i:Inter Area 1:External Type 1 2:External Type 2
STATIC Codes - d:DHCPv6
        Type IPv6 Prefix           Next Hop Router    Interface     Dis/Metric     Uptime src-vrf
1       C    2001:1::/32           ::                 ve 201        0/0            17m36s -
2       S    2002:1::/32           ::                 ve 201        1/1            13m55s -
                                   fe80::1            ve 201                               -
```

   The example shows a directly connected and a static route in the IPv6 routing table. Both routes are through virtual interface (ve) 201.

3. To display the IPv6 route table information, enter the **show ipv6 route** command.

```
device# show ipv6 route
IPv6 Routing Table for VRF "default-vrf"
Total number of IPv6 routes: 11
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

1200:1201::/64, attached
    *via ::, Eth 2/45, [0/0], 45m29s, direct, tag 0
1200:1201::1:1/128, attached
    *via ::, Eth 2/45, [0/0], 45m29s, local, tag 0
1200:1202::/64, attached
    *via ::, Ve 2, [0/0], 45m26s, direct, tag 0
1200:1202::1:1/128, attached
    *via ::, Ve 2, [0/0], 45m26s, local, tag 0
2221::/32
    *via 1200:1201::1:2, Eth 2/45, [100/10], 11m41s, static, tag 300
2222::/48
    *via fe80::205:33ff:fee6:a531, Eth 2/45, [1/1], 43m44s, static, tag 0
2222::1/128
    *via fe80::205:33ff:fee6:a531, Eth 2/45, [110/1], 0m7s, ospfv3, intra, tag 0
2223::/64
    *via 1200:1202::1:2, Ve 2, [1/1], 3m45s, static, tag 0
2224::1/128
    *via fe80::205:33ff:fee6:a501, Ve 2, [1/1], 43m41s, static, tag 0
fe80::/10, attached
    *via ::, , [0/0], 6h30m, local, tag 0
ff00::/8, attached
    *via ::, Null0, [0/0], 6h30m, local, tag 0
```

In the IPv6 route table, connected, static, RIP, OSPF, and BGP routes are listed, along with the destination address, the next hop router, the interface used toward the destination, and the administrative distance and cost for each route.

4. To display the IPv6 route table information, enter the **show ipv6 route** command.

```
device# show ipv6 route
IPv6 Routing Table for VRF "default-vrf"
Total number of IPv6 routes: 11
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

1200:1201::/64, attached
    *via ::, Eth 0/40, [0/0], 45m29s, direct, tag 0
1200:1201::1:1/128, attached
    *via ::, Eth 0/40, [0/0], 45m29s, local, tag 0
1200:1202::/64, attached
    *via ::, Ve 2, [0/0], 45m26s, direct, tag 0
1200:1202::1:1/128, attached
    *via ::, Ve 2, [0/0], 45m26s, local, tag 0
2221::/64
    *via 1200:1201::1:2, Eth 0/40, [100/10], 11m41s, static, tag 300
2222::/64
    *via fe80::205:33ff:fee6:a531, Eth 0/40, [1/1], 43m44s, static, tag 0
2222::1/128
    *via fe80::205:33ff:fee6:a531, Eth 0/40, [110/1], 0m7s, ospfv3, intra, tag 0
2223::/64
    *via 1200:1202::1:2, Ve 2, [1/1], 3m45s, static, tag 0
2224::1/128
    *via fe80::205:33ff:fee6:a501, Ve 2, [1/1], 43m41s, static, tag 0
fe80::/10, attached
    *via ::, , [0/0], 6h30m, local, tag 0
ff00::/8, attached
    *via ::, Null0, [0/0], 6h30m, local, tag 0
```

In the IPv6 route table, connected, static, RIP, OSPF, and BGP routes are listed, along with the destination address, the next hop router, the interface used toward the destination, and the administrative distance and cost for each route.

5. To narrow the display to static routes, enter the **show ipv6 static route** command.

```
device# show ipv6 static route
IPv6 Configured Static Routes for VRF "default-vrf"

3002:7::/64-> 1200:3::1:2 preference: 1
   nh_vrf (default-vrf)

3002:9::/64-> 1200:4::1:2 preference: 1
   nh_vrf (default-vrf)
device#
```

The example shows two IPv6 static routes in the default VRF.

DHCPv4

- DHCPv4 overview 139
- IP DHCP Relay function 139
- Configuring IP DHCP Relay 141
- DHCP relay agent supported user scenarios 143
- DHCP relay agent information option 82 144
- Enabling the DHCP Relay Agent Information option 145
- Configuring DHCP relay gateway address 146
- Configuring DHCP relay gateway interface 147
- VRF support 147
- Displaying IP DHCP Relay statistics 148
- Displaying IP DHCP Relay addresses on specific devices 150
- Displaying IP DHCP relay addresses for an interface 153
- Clearing IP DHCP relay statistics 154

DHCPv4 overview

The Dynamic Host Configuration Protocol for DHCPv4 enables DHCP servers to pass configuration parameters such as IPv4 addresses to IPv4 hosts.

The Dynamic Host Configuration Protocol (DHCP) is based on the Bootstrap Protocol (BOOTP) and provides configuration parameters such as IP addresses, default routes, DNS server addresses, access control, QoS policies, and security policies stored in DHCP server databases to DHCP clients upon request. DHCP enables the automatic configuration of client systems. DHCP removes the need to configure devices individually. Clients can set network properties by connecting to the DHCP server instead. This protocol consists of two components; a protocol to deliver host-specific configuration parameters from a DHCP server to a host, and a mechanism to allocate network addresses to hosts.

DHCP is built on a client-server model, where designated DHCP server hosts allocate network addresses and deliver configuration parameters to dynamically configured hosts.

You can enable DHCP service on VDX switches so that they can automatically obtain an Ethernet IP address, prefix length, and default gateway address from the DHCP server. Refer to the "Configuring an IPv4 address with DHCP" section of the *Extreme Network OS Management Configuration Guide* for more information.

You can enable DHCP service on a router so that they can automatically obtain an Ethernet IP address, prefix length, and default gateway address from the DHCP server.

IP DHCP Relay function

DHCP relays are an important feature for large networks as they allow communication between DHCP servers and clients located on different subnets.

In small networks with only one IP subnet, DHCP clients can communicate directly with DHCP servers. Clients located on a different subnet than the DHCP server cannot communicate with that server without obtaining an IP address with appropriate routing information.

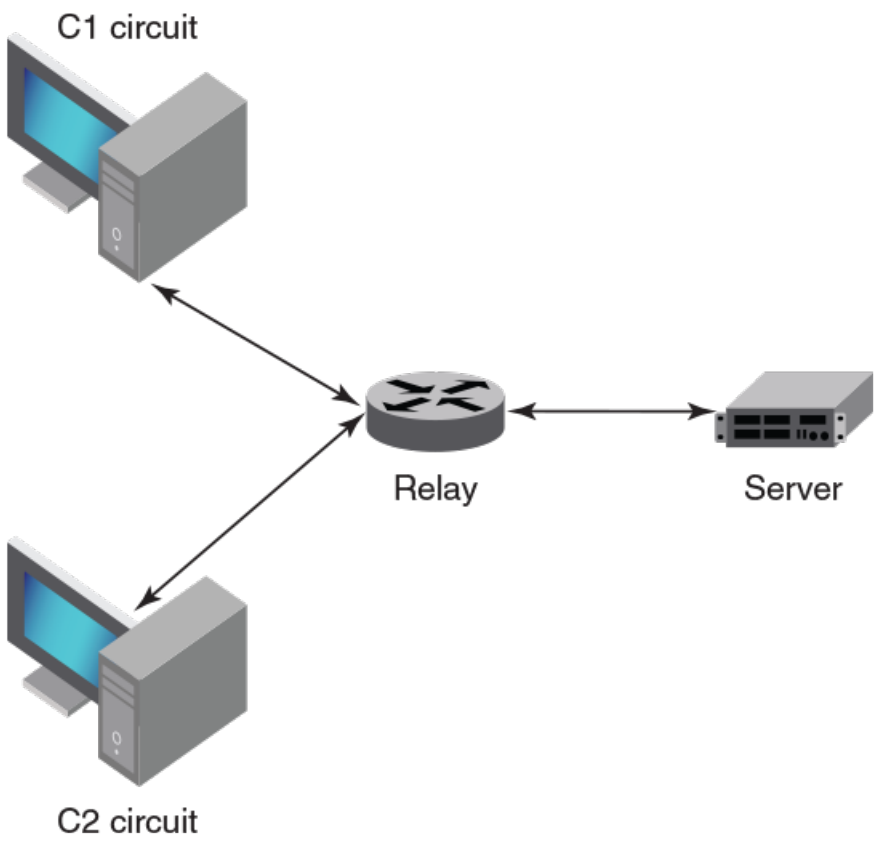
By installing a DHCP relay agent on different subnets in a large network, broadcast DHCP packets can be forwarded from a DHCP client to locate a DHCP server on a remote subnet. The relay agent's IP address is stored in the gateway IP address (GIADDR) field of the DHCP packet, The DHCP server uses the GIADDR field to find the subnet where the relay agent received the broadcast, and then

assigns IP addresses to that subnet. The DHCP server replies to the client with a unicast message to the GIADDR address and the relay agent will forward the response to the local network.

By installing a DHCP relay agent on different subnets in a large network, broadcast DHCP packets can be forwarded from a DHCP client to locate a DHCP server on a remote subnet. The relay agent's IP address is stored in the gateway IP address (GIADDR) field of the DHCP packet, The DHCP server uses the GIADDR field to find the subnet where the relay agent received the broadcast, and then assigns IP addresses to that subnet. The DHCP server replies to the client with a unicast message to the relay agent's IP address and the relay agent will forward the response to the local network.

If a relay agent is configured on a Ve interface, Ve 100, which acts over two physical interfaces, ethernet 1/1 and ethernet 1/2, corresponding to two client circuit C1 and C2, then broadcast replies from the server will be forwarded to both the circuits, irrespective of the circuit from which the client packet is received by the relay agent, when both the circuits belong to the same broadcast network.

FIGURE 16 IP DHCP relay function



In the above scenario, C1 is connected to Eth 1/1 and C2 is connected to Eth 1/2 physical interfaces. The relay agent is configured at Ve 100. Ve 100 is configured on VLAN 100, where Eth 1/1 and 1/2 ports are tagged. In such a configuration, broadcast responses from the server will be received by both C1 and C2.

Configuring IP DHCP Relay

Configure the IP DHCP Relay agent on any Layer 3 interface using the IP address of the DHCP server where client requests are to be forwarded.

Layer 3 interfaces can be a virtual Ethernet (VE), port channel (PO), or a physical 1, 10, or 40 Gigabit Ethernet interface.

Layer 3 interfaces can be a virtual Ethernet (VE), or physical interfaces.

You can configure the IP DHCP Relay agent using the **ip dhcp relay address** command followed by the IP address of the DHCP server. Use the **use-vrf** *vrf-name* parameter if the DHCP server and client interface are on different Virtual Forwarding and Routing (VRF) instances.

The following are considerations and limitations when configuring the IP DHCP Relay agent:

- You can configure up to sixteen DHCP server IP addresses per interface. When multiple addresses are configured, the relay agent relays the packets to all server addresses. The total number of addresses configurable on the router is 4000.
- You can configure up to four DHCP server IP addresses per interface. When multiple addresses are configured, the relay agent relays the packets to all server addresses.
- The DHCP server and clients it communicates with can be attached to different VRF instances. When clients and the DHCP server are on different VRFs, use the **use-vrf** *vrf-name* option with the **ip dhcp relay address** command, where *vrf-name* is the VRF where the DHCP server is located. For more information on VRF support for the IP DHCP Relay, refer to VRF support on page 147.

Perform the following steps to configure an IP DHCP Relay:

1. In privileged EXEC mode, enter the **configure terminal** command to enter the global configuration mode.

   ```
   device# configure
   ```

2. Enter the rbridge-id command with an RBridge ID to enter the RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 1
   ```

3. Enter the **interface** command followed by the interface ID to enter the interface configuration mode for the Ve, PO or physical interface where you want to configure the IP DHCP Relay.

   ```
   device(config-rbridge-id-1)# int Ve 101
   ```

4. Enter the interface configuration mode.

   ```
   device(config)# interface ethernet 1/18
   ```

5. Enter the interface configuration mode.

   ```
   device(config)# interface ethernet 0/1
   ```

6. Enter the IP address of the DHCP server.

   ```
   device(config-if-eth-1/18)# ip dhcp relay address 100.1.1.2
   ```

7. Enter the IP address of the DHCP server.

   ```
   device(conf-if-eth-0/1)# ip dhcp relay address 100.1.1.2
   ```

8. Enter the **ip dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command where *ip-addr* is the IP address of the DHCP server. Use the **use-vrf** *vrf-name* option if the DHCP server is on a different VRF instance than the interface where the client is connected.

```
device(config)# ip dhcp relay address 100.1.1.2
```

9. Enter the **ip dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command where *ip-addr* is the IP address of the DHCP server. Use the **use-vrf** *vrf-name* option if the DHCP server is on a different VRF instance than the interface where the client is connected.

```
(config)# int ve 100
device(config-Ve-100)# ip dhcp relay 100.1.1.2 use-vrf blue
```

10. To remove the IP DHCP Relay address enter the **no ip dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command.

The following is an example of configuring two IP DHCP Relay addresses on a physical 1 GbE interface in slot 2, port 4 on RBridge ID 1.

> **NOTE**
> In this example, the local DHCP server IP address is 10.1.1.2.

```
device# config
Entering configuration mode terminal

device(config)# rbridge-id 1
device(config-rbridge-id-1)# int Ve 101
device(config-Ve-101)# ip dhcp relay address 100.1.1.2
device(config-Ve-101)# ip dhcp relay address 12.3.4.6
device(config-Ve-101)# ip dhcp relay address 10.1.1.2
```

The following is an example of configuring two IP DHCP Relay addresses on virtual Ethernet interface 102.

> **NOTE**
> In this example, the IP address 3.1.1.255 is the local subnet broadcast address. The relay agent relays the DHCP packets to the directed broadcast address and all addresses for DHCP servers configured on the interface.

```
device# config
Entering configuration mode terminal

device(config)# rbridge-id 2
device(config-rbridge-id-2)# int Ve 102
device(config-Ve-102))# ip dhcp relay address 200.1.1.2
device(config-Ve-102)# ip dhcp relay address 10.1.1.255
```

The following example removes an IP DHCP Relay address using the **no** option in the **ip dhcp relay address** *ip-addr* command:

```
device(config-if)# no ip dhcp relay address 200.1.1.2
```

If the DHCP server is on a different VRF than the interface where the client is connected, use the **use-vrf** *vrf-name* option in the **ip dhcp relay address** *ip-addr* command.

> **NOTE**
> If the **use-vrf** *vrf-name* option is not used, it is assumed that the DHCP server and client interface are on the same VRF.

```
device# config
Entering configuration mode terminal

device(config)# rbridge-id 2
device(config-rbridge-id-2)# int Ve 103
device(config-Ve-103)# ip dhcp relay address 10.1.2.255 use-vrf blue
```

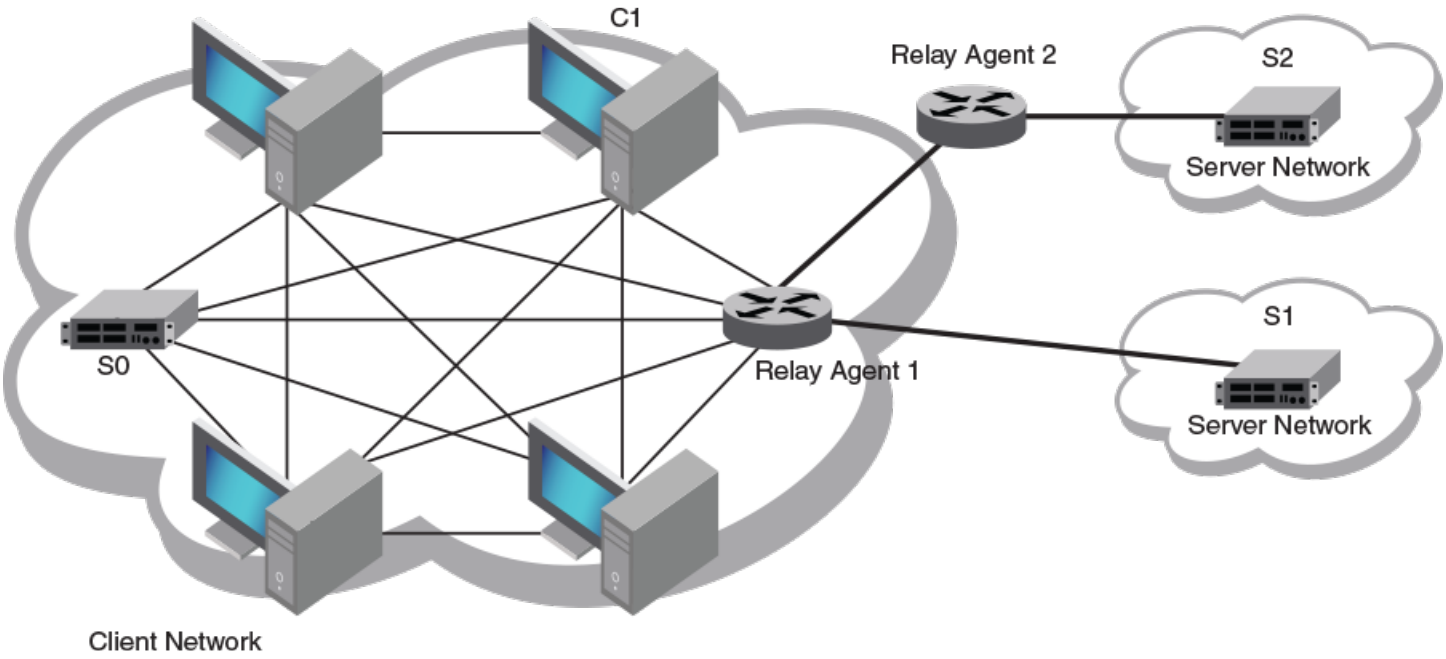
The following example removes an IP DHCP Relay address using the **no** option in the **ip dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command:

```
device(config-ve-103)# no ip dhcp relay address 10.1.2.255 use-vrf blue
```

DHCP relay agent supported user scenarios

The following examples illustrate the network environments where the DHCP relay agent is supported.

FIGURE 17 DHCP relay agent user scenarios



DHCP server local

Client 1 and Server 0 (S0) are on the same subnet (same broadcast domain). A relay agent is not required in this scenario, because the server will receive broadcast discover messages from the client directly; a configured relay agent in this scenario is redundant, as the relay agent would relay (unicast) the received DHCP packet again to the destination server or client, and as a result both parties receive multiple packets.

DHCP server remote

Client 1 (C1) and Server 1 (S1) are on different subnets but directly linked through a relay agent.

DHCP servers in a network

Client 1 (C1) and Server 2 (S2) are on different subnets and connected through relay chaining (relay multi-hops).

Multiple DHCP servers

Server 1 (S1) and Server 2 (S2) are two server addresses on different subnets configured at relay agent 1.

DHCP server in different VRFs

Client 1 (C1) and Server 1 (S1) are on different VRFs.

DHCP relay agent information option 82

When DHCP relay agent information option 82 is enabled, relay agent information option-82 is inserted by the relay agent before relaying DHCP client requests to the server.

The relay agent information option-82 allows the DHCP server to select a sub-range in the DHCP server address pool. The DHCP server echos the option 82 in the DHCP reply packet. The DHCP relay agent validates and removes the option 82 information, and then sends the response to the DHCP client.

Adding option 82 to the DHCP client helps address the following security issues:

- Allows the relay agent to identify the circuit to which to forward replies.
- Prevents DHCP IP address exhaustion attacks. IP address exhaustion occurs when an attacker requests all available IP addresses from a DHCP server by sending requests with fake client MAC addresses.
- Prevents permanently assigning an IP address to a particular user or modem.
- Prevents spoofing of client identifier fields used to assign IP addresses.
- Prevents denial of service (DoS) attacks.

The DHCP Relay Agent Information Option is a container option for specific agent-supplied sub-options. The format of the relay agent information option is as follows:

Code	Len	Agent Information Field					
82	N	i1	i2	i3	i4	...	iN

NOTE
The length N represents the total number of octets in the Agent Information Field. The Agent Information field consists of a sequence of SubOpt/Length/Value tuples for each sub-option.

Relay Agent Circuit ID sub-option

Sub-option type (1 byte)	Length (1 byte)	VLAN ID <string> (4 bytes)	IF-description string (4 bytes)
2	68		

NOTE
The circuit ID is a combination of the VLAN-ID and the interface description string. If the interface description is not configured, the default string " Extremenetworks" is used in the circuit ID.

Agent remote ID sub-option

Sub-option type (1 byte)	Length (1 byte)	VLAN ID (2 bytes)	MAC address (6 bytes)
2	8		

Relay agent operation with Option 82 enabled

When Option 82 is enabled, the relay agent performs the following actions:

- If the client receives a DHCP packet with the GIADDR field set to zero, but with the Option 82 already present, the relay agent discards the packet and increments the error count.
- If the client receives a DHCP packet with the GIADDR field set to a GIADDR implemented by the local agent, the packet is discarded.
- Adds the IP address of the relay agent (in the GIADDR field).
- Inserts the Option 82 information as the last option in a request packet. Option 82 information contains the remote ID sub-option and the circuit ID sub-option.
- Relays the packet to the DHCP server.
- Removes Option 82 from the received packets from the DHCP server after validation.
- Forwards the packet to the client.

Configuration considerations

The following configuration considerations apply to DHCP Option 82:

- If the relay agent is configured over a Ve interface, the remote-id will be the ifindex of the Ve interface, and the broadcast replies from the server are flooded to all the tagged interfaces configured in the Ve.
- The relay agent does not monitor the client requests during the renewal phase. Also, the device forwards request packets with a non-zero GIADDR from a different relay agent.
- All client interfaces are treated as untrusted interfaces.
- You cannot configure each sub-option separately. Enabling Option 82 enables the insertion of the circuit ID and remote ID sub-options.
- DHCP relay Option 82 can be enabled or disabled globally. You cannot enable or disable this option at the interface level.

Enabling the DHCP Relay Agent Information option

Complete the following steps to enable insertion or removal of DHCP relay information option-82 present in the DHCP client and server packets respectively.

1. Enter the global configuration mode.

```
device# configure
```

2. Enter the rbridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **ip dhcp relay information option** command to enable option-82.

```
device(config)# ip dhcp relay information option
```

4. Enter the **ip dhcp relay information option** command to enable option-82.

```
device(config-rbridge-id-1)# ip dhcp relay information option
```

5. Specify the interface description (string) to be used for the IPv4 DHCP relay Option 82 Circuit ID for a Ve interface.

```
device(config-Vlan-1)# description dcip
```

6. You can also specify the interface description (string) to be used for the IPv4 DHCP relay Option 82 Circuit ID for a physical router port.

```
device(conf-if-te-201/0/7)# description dcip
```

Configuring DHCP relay gateway address

When the DHCP relay agent forwards a BootP or DHCP request, the relay agent stamps the Gateway Address field.

The default value the relay agent uses to stamp the packet is the lowest-numbered primary IP address configured on the interface that received the request.

The following considerations apply to DHCP relay gateway address configuration:

- If the DHCP relay gateway address is configured, but no IP address is configured on the interface, the DHCP relay agent will not relay the requests until IP address is configured on the interface.
- When the IP DHCP relay gateway address is configured, but does not match the IP addresses configured on the interface, the DHCP relay agent uses the default value, the lowest-numbered IP address configured on the interface. When the gateway address is one of the IP addresses configured on the interface, it will be used to stamp the requests.
- Virtual IP addresses or local subnet broadcast addresses must not be used as the DHCP relay gateway address.
- Any primary or secondary IP address configured on the interface can be used as the DHCP relay gateway address.

The following steps change the DHCP relay gateway address used for stamping BootP or DHCP requests received on a Layer 3 interface.

1. Enter the global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode.

```
device(config)# interface ethernet 1/4
```

3. Enter the interface configuration mode.

```
device(config)# interface ethernet 0/1
```

4. Enter the **ip dhcp relay gateway** command followed by the IP address used for stamping BootP or DHCP requests received on a Layer 3 interface.

```
device(config-if-eth-1/4)# ip dhcp relay gateway 10.50.22.26
```

   The relay agent places the IP address 10.50.22.26 in the Gateway Address field of BootP or DHCP requests that the router receives on port 1/4 and forwards to the BootP or DHCP server.

5. Enter the **ip dhcp relay gateway** command followed by the IP address used for stamping BootP or DHCP requests received on a Layer 3 interface.

```
device(conf-if-eth-0/1)# ip dhcp relay gateway 10.50.22.26
```

   The relay agent places the IP address 10.50.22.26 in the Gateway Address field of BootP or DHCP requests that the router receives on port 0/1 and forwards to the BootP or DHCP server.

Configuring DHCP relay gateway interface

When using this configuration, the GIADDR continues to be present but only relays the IP address that is to be used by the DHCP server to communicate with.

The following considerations apply to the DHCP relay gateway interface configuration:

- The Option-82 must be enabled globally along with the gateway address configuration in the relay agent.
- The DHCP server must be configured to support the GIADDR and Option-82 link-selection sub-option.
- The DHCP relay gateway address must be already configured.
- Only one DHCP relay gateway can be configured on an interface.

The following steps set the IP address configured in the interface as the DHCP relay gateway used for DHCP requests received on a Layer 3 interface.

1. Enter the global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the interface configuration mode.

   ```
   device(config)# interface ethernet 0/1
   ```

3. Enter the **ip dhcp relay gateway interface** command.

   ```
   device(conf-if-eth-0/1)# ip dhcp relay gateway interface ve 101
   ```

   The relay agent places the IP address set in the interface named ve 101 in the Gateway Address field of DHCP requests that are received on port 0/1 and forwards to the DHCP server.

VRF support

Virtual Routing and Forwarding (VRF) is a technology that controls information flow within a network by partitioning the network into different logical VRF domains to isolate traffic. This allows a single device to have multiple containers of routing tables or Forwarding Information Bases (FIBs), with one routing table for each VRF instance. This permits a VRF-capable router to function as a group of multiple virtual routers on the same physical router.

Inter-VRF route leaking allows leaking of specific route prefixes from one VRF instance to another on the same physical router, which eliminates the need for external routing.

In a DHCP setting, route leaking is controlled through a single DHCP server (which may be on a different VRF). This permits multiple VRFs to communicate with that server.

The IP DHCP Relay is supported in configurations where the DHCP server is on the same or different VRF instances than the interface through which the client is connected. When the DHCP server and client are on different VRFs, this is called inter-VRF deployment. For inter-VRF deployment, use the use-vrf *vrf-name* option with the **ip dhcp relay address** command, where *vrf-name* is the VRF where the DHCP server is located.

For more information on VRFs, refer to the Virtual Routing and Forwarding chapter.

Supported VRF configuration examples

Following are examples of VRF configurations that are supported for IP DHCP Relay:

- Client interface and DHCP server are on the same VRF instance. For example:
  - VE interface 100 in VRF "red"
  - IP address of interface - 3.1.1.1/24
  - IP DHCP Relay address (20.1.1.2) in VRF "red"
  - IP DHCP Relay address (20.1.1.2)
- Client interface and DHCP servers are on different VRF instances. For example:
  - VE interface 100 in default VRF
  - IP address of interface - 3.1.1.1/24
  - IP DHCP Relay address (100.1.1.2) in VRF "blue"
  - IP DHCP Relay address (1.2.3.4.6) in VRF "red"

  A maximum of 128 of these inter-VRF IP DHCP Relay address configurations is allowed per node. A VRF route leak configuration is required for these configurations. In the preceding example, a VRF route leak configuration is required on the default VRF as follows:

  - ip route 100.1.1.2/32 next-hop-vrf blue 100.1.1.2
  - ip route 100.1.1.2/32 next-hop-vrf blue <exit interface/next-hop-ip>
  - ip route 12.3.4.6/32 next-hop-vrf red 14.3.4.6
  - ip route 12.3.4.6/32 next-hop-vrf red <exit interface/next-hop-ip>

VRF configuration examples to avoid

The following examples of VRF configurations are not recommended for IP DHCP Relay.

- The same IP DHCP Relay address configured on different VRFs. As an example:
  - VE interface 100 in default VRF
  - IP address of interface - 3.1.1.1/24
  - IP DHCP Relay address (30.1.1.2) in VRF "blue"
  - IP DHCP Relay address (30.1.1.2) in VRF "red"
- The same IP DHCP Relay address configured on two interfaces with the same address, and both interfaces are on different VRFs. As an example:
  - **VE interface 100 in default VRF**
  - IP address of interface - 3.1.1.1/24
  - IP DHCP Relay address (20.1.1.2) in VRF "blue"
  - **VE interface 200 in VRF "blue"**
  - IP address of interface - 3.1.1.1/24
  - IP DHCP Relay address (20.1.1.2) in VRF "blue"

Displaying IP DHCP Relay statistics

Display information about the DHCP Relay function, such as the DHCP Server IP address configured on the device and the number of DHCP packets received and dropped by the interface configured for IP DHCP Relay.

Use the **show ip dhcp relay statistics** command to display the following information about the IP DHCP Relay function:

- DHCP Server IP Address configured in the device.

- Number of DHCP DISCOVERY, OFFER, REQUEST, ACK, NAK, DECLINE, and INFORM packets received.
- Number of DHCP DISCOVERY, OFFER, REQUEST, ACK, NAK, DECLINE, INFORM, and RELEASE packets received.
- Number of DHCP client packets received (on port 67) and relayed by the Relay Agent.
- Number of DHCP server packets received (on port 68) and relayed by the Relay Agent.
- Number of DHCP client packets dropped by the Relay Agent.
- Number of DHCP server packets dropped by the Relay Agent.

NOTE
You can specify a list of RBridge IDs separated by commas, or a range separated by a dash (for example, 1-2). No spaces are allowed in the range string. The range does not need to be contiguous (for example, 1-2,5). You can also specify "all" for all RBridge IDs in the logical chassis cluster.

1. Access a device where an IP DHCP Relay has been configured on an interface.
2. In privileged EXEC mode, enter **show ip dhcp relay statistics**. Optionally, you can specify specific RBridge IDs if you only want to view the statistics for those RBridges.
3. In privileged EXEC mode, enter **show ip dhcp relay statistics**.

The following example displays statistics on a local switch.

```
device# show ip dhcp relay statistics


                DHCP Relay Statistics - RBridge Id: 3
                --------------------------------------
Address   Disc.   Offer  Req.   Ack   Nak   Decline   Release   Inform
-------   -----   -----  ----   ---   ---   -------   -------   ------
10.1.0.1    400     100  2972  2968     0         0         0        0
20.2.0.1    400     100  2979  2975     0         0         0        0
30.3.0.1    400     100  3003  2998     0         0         0        0
40.4.0.1    400     100  3026  3018     0         0         0        0

Client Packets: 12780
Server Packets: 12359
Client Packets Dropped: 0
Server Packets Dropped: 0
```

The following example displays statistics for a cluster with RBridge 1 and RBridge 3.

```
device# show ip dhcp relay statistics rbridge-id 1,3
DHCP Relay Agent Information Option Enabled

                DHCP Relay Statistics - RBridge Id: 1
                ----------------------------------------
Address   Disc.   Offer  Req.   Ack   Nak   Decline   Release   Inform
-------   -----   -----  ----   ---   ---   -------   -------   ------
2.3.4.5     300     100  1211  1201     0         0         0        0
10.0.1.2    300     100  1211  1207     0         0         0        0

Client Packets: 2701
Server Packets: 2932
Client Packets Dropped: 0
Server Packets Dropped: 0

                DHCP Relay Statistics - RBridge Id: 3
                ----------------------------------------
Address   Disc.   Offer  Req.   Ack   Nak   Decline   Release   Inform
-------   -----   -----  ----   ---   ---   -------   -------   ------
10.0.0.5      0       0     0     0     0         0         0        0
10.0.1.2      0       0     0     0     0         0         0        0

Client Packets: 0
Server Packets: 0
Client Packets Dropped: 0
Server Packets Dropped: 0
```

The following example displays statistics for the device.

```
device# show ip dhcp relay statistics

Address   Disc.   Offer  Req.   Ack   Nak   Decline     Inform
-------   -----   -----  ----   ---   ---   -------     ------
2.3.4.5     300     100  1211  1201     0         0          0
10.0.1.2    300     100  1211  1207     0         0          0

Client Packets: 2701
Server Packets: 2932
Client Packets Dropped: 0
Server Packets Dropped: 0
```

Displaying IP DHCP Relay addresses on specific devices

Use the show ip dhcp relay address command to display all IP DHCP Relay addresses configured on specific switches (specified by RBridge IDs) in a logical chassis cluster. You can use the **all** keyword to display configured addresses on all RBridge IDs in a cluster.

Use the **show ip dhcp relay address** command to display all IP DHCP Relay addresses configured on specific devices.

1. Access a device where an IP DHCP Relay has been configured.
2. In privileged EXEC mode, enter the **show ip dhcp relay address rbridge-id** *rbridge-id* command.
3. In privileged EXEC mode, enter the **show ip dhcp relay address** command.

The following example displays addresses configured on the interfaces of the device.

```
device# show ip dhcp relay address
DHCP Relay Information Option: Enabled

---------                    -------------                --------
Interface                    Relay Address                VRF Name
---------                    -------------                --------
eth1/18                      105.0.0.10                   default-vrf
eth1/18                      107.0.0.1                    default-vrf
ve 200                       107.0.0.1                    default-vrf
```

The following example displays addresses configured on the interfaces of the device.

```
device# show ip dhcp relay address
DHCP Relay Information Option: Enabled

---------                    -------------                --------
Interface                    Relay Address                VRF Name
---------                    -------------                --------
eth0/18                      105.0.0.10                   default-vrf
eth0/18                      107.0.0.1                    default-vrf
ve 200                       107.0.0.1                    default-vrf
```

The following is an example of displaying addresses configured on interfaces of a local switch. Notice that the RBridge ID is not needed in the command.

```
device# show ip dhcp relay address
DHCP Relay Information Option: Disabled

                             Rbridge Id:    2
                             ----------------
Interface                    Relay Address                VRF Name
---------                    -------------                --------
Te 2/0/35                    10.5.1.1                     default-vrf
Te 2/0/35                    10.3.4.5                     default-vrf
Ve 300                       10.0.1.2                     default-vrf
Ve 300                       10.0.0.5                     default-vrf
```

The following is an example of displaying addresses configured on interfaces on a specific RBridge.

```
device# show ip dhcp relay address rbridge-id 53
DHCP Relay Information Option: Disabled

                             Rbridge Id:   53
                             ----------------
Interface                    Relay Address                VRF Name
---------                    -------------                --------
Ve 300                       20.0.1.2                     blue
Ve 300                       30.1.1.5                     green
Ve 300                       40.2.1.1                     default-vrf
```

The following is an example of displaying addresses configured on interfaces on all RBridge IDs in a logical chassis cluster.

```
device# show ip dhcp relay address rbridge-id all
DHCP Relay Information Option: Disabled

                              Rbridge Id:   53
                              ----------------
Interface                     Relay Address                VRF Name
---------                     -------------                --------
Ve 300                        20.0.1.2                     blue
Ve 300                        30.1.1.5                     green
Ve 300                        40.2.1.1                     default-vrf
DHCP Relay Information Option: Disabled

                              Rbridge Id:    2
                              ----------------
Interface                     Relay Address                VRF Name
---------                     -------------                --------
Te 2/0/35                     10.5.1.1                     default-vrf
Te 2/0/35                     10.3.4.5                     default-vrf
Ve 300                        10.0.1.2                     default-vrf
Ve 300                        10.0.0.5                     default-vrf
```

The following example displays configured IPv4 DHCP Relay addresses on a specified physical interface.

```
device# show ip dhcp relay address interface tengigabitethernet 2/0/37
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:0000:0027f8cad501
Rbridge Id: 2
----------------
Interface Relay Address VRF Name
--------- ------------- --------
Te 2/0/37 8.8.8.8 default-vrf
DHCP relay information option 13
```

The following example displays configured IPv4 DHCP Relay addresses on a specified port channel and RBridge.

```
device# show ip dhcp relay address interface port-channel 10 rbridge-id 2
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:0000:0027f8cad51d
Rbridge Id: 2
----------------
Interface Relay Address VRF Name
--------- ------------- --------
Po 10 20.1.1.1 default-vrf
```

The following example displays configured IPv4 DHCP relay addresses on a VE interface and RBridge.

```
device# show ip dhcp relay address interface ve 10 rbridge-id 2
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:000a:0027f8cad4d9
Rbridge Id: 2
----------------
Interface Relay Address VRF Name
--------- ------------- --------
Ve 10 4.4.4.4 default-vrf
```

The following example displays configured IPv4 DHCP relay addresses on a VE interface and RBridge.

```
device# show ip dhcp relay address interface ve 10 rbridge-id 51,53
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:000a:0027f852c94e
Rbridge Id: 51
----------------
Interface Relay Address VRF Name
--------- ------------- --------
Ve 10 4.4.4.4 default-vrf
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:000a:0027f8cb1377
Rbridge Id: 53
----------------
Interface Relay Address VRF Name
--------- ------------- --------
Ve 10 14.1.1.1 default-vrf
```

Displaying IP DHCP relay addresses for an interface

You can display IP DHCP relay addresses configured on specific interfaces of a device.

To display the IP DHCP relay addresses configured for an interface, use the **show ip dhcp relay address interface** command followed by the interface ID to display IP DHCP relay addresses configured on a specific interface.

1. Access a device where an IP DHCP relay has been configured on an interface.
2. In privileged EXEC mode, enter the **show ip dhcp relay address interface** command followed by the interface ID.

The following example displays addresses configured on interface 1/18.

```
device# show ip dhcp relay address interface ethernet 1/18
DHCP Relay Information Option: Enabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:0000:000000f8003e
---------                     -------------                 --------
Interface                     Relay Address                 VRF Name
---------                     -------------                 --------
eth 1/18                       105.0.0.10                    default-vrf
eth 1/18                       107.0.0.1                     default-vrf
```

The following example displays addresses configured on interface 0/24.

```
device# show ip dhcp relay address interface eth 0/24

Interface                        Relay Address                   VRF Name
---------                     -------------                       --------
eth 0/24                        10.3.4.5                             Blue
eth 0/24                              10.5.1.1                                        Blue
```

The following example displays addresses configured on VE 200.

```
device# show ip dhcp relay address interface ve 200
DHCP Relay Information Option: Enabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:0000:000000f8003d
---------                     -------------                 --------
Interface                     Relay Address                 VRF Name
---------                     -------------                 --------
ve 200                         107.0.0.1                     default-vrf
```

The following is an example for a 10 GbE interface of a local switch.

```
device# show ip dhcp relay address interface te 2/0/35
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:0000:0027f8cad4ff
                              Rbridge Id:    2
                              ----------------
Interface                     Relay Address                 VRF Name
---------                     -------------                 --------
Te 2/0/35                     10.5.1.1                      default-vrf
Te 2/0/35                     10.3.4.5                      default-vrf
```

The following is an example for a VE interface on a specific switch (RBridge ID 2).

```
device# show ip dhcp relay address int ve 300 rb 2
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:012c:0027f8cad4d9
                              Rbridge Id:    2
                              ----------------
Interface                     Relay Address                 VRF Name
---------                     -------------                 --------
Ve 300                        10.0.1.2                      default-vrf
Ve 300                        10.0.0.5                      default-vrf
```

The following is an example for displaying addresses on for a specific VE interface on a range of switches (specified by RBridge IDs) in a logical chassis cluster.

> NOTE
> You can specify a list of RBridge IDs separated by commas, or a range separated by a dash (for example, 1-2). No spaces are allowed in the range string. The range does not need to be contiguous (for example, 1-2,5). You can also specify **all** for all RBridge IDs in the logical chassis cluster.

```
device# show ip dhcp re address int ve 300 rbridge-id 2,53
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:012c:0027f8cad4d9
                              Rbridge Id:    2
                              ----------------
Interface                     Relay Address                 VRF Name
---------                     -------------                 --------
Ve 300                        10.0.1.2                      default-vrf
Ve 300                        10.0.0.5                      default-vrf
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:012c:0027f8cb1377
                              Rbridge Id:   53
                              ----------------
Interface                     Relay Address                 VRF Name
---------                     -------------                 --------
Ve 300                        20.0.1.2                      blue
Ve 300                        30.1.1.5                      green
Ve 300                        40.2.1.1                      default-vrf
```

Clearing IP DHCP relay statistics

Use the **clear ip dhcp relay statistics** command to clear all IP DHCP Relay statistics for specific relay IP addresses, for addresses on specific RBridge IDs, or all addresses for RBridge IDs in a logical chassis cluster.

Use the **clear ip dhcp relay statistics** command to clear all IP DHCP Relay statistics for specific relay IP addresses.

For command parameters you can specify the IP DHCP Relay address and RBridge IDs. You can specify a list of RBridge IDs separated by commas, or a range separated by a dash (for example, 1-2). No spaces are allowed in the range string. The range does not need to be contiguous (for example, 1-2,5). You can also specify “all” for all RBridge IDs in the logical chassis cluster.

1. Access a device where an IP DHCP Relay has been configured on an interface.

2. In privileged EXEC mode, issue the **clear ip dhcp relay statistics ip-address** *ip-address* **rbridge-id** *rbridge-id*.
3. In privileged EXEC mode, issue the **clear ip dhcp relay statistics ip-address** *ip-address*.

The following command clears statistics for IP DHCP Relay address 10.1.0.1 configured on RBridges 1, 3, and 5.

```
device# clear ip dhcp relay statistics ip-address 10.1.0.1 rbridge-id 1,3,5
```

The following command clears statistics for all IP DHCP Relay addresses on RBridges 1, 3, and 5.

```
device# clear ip dhcp relay statistics rbridge-id 1,3,5
```

The following command clears statistics for IP DHCP Relay address 105.0.0.10.

```
device# clear ip dhcp relay statistics ip-address 105.0.0.10
```

DHCPv6

- DHCPv6 overview ... 157
- DHCP relay agent for IPv6 ... 157
- DHCPv6 relay agent ... 157
- DHCPv6 multicast addresses and UDP ports ... 158
- DHCPv6 address assignment ... 159
- DHCPv6 message format ... 160
- DHCPv6 relay provisioning ... 161
- Configuring IPv6 DHCP relay ... 162
- Displaying DHCPv6 relay addresses on specific devices ... 163
- Displaying DHCPv6 relay addresses for an interface ... 165
- Displaying IPv6 DHCP relay statistics ... 166
- Clearing IP DHCPv6 relay statistics ... 168

DHCPv6 overview

The Dynamic Host Configuration Protocol for IPv6 (DHCP) enables DHCP servers to pass configuration parameters such as IPv6 network addresses to IPv6 nodes.

The DHCPv6 protocol offers the capability of automatic allocation of reusable network addresses and additional configuration flexibility.

On NetIron devices you can configure DHCP relay agent, and the DHCP Prefix Delegation.

On Network OS devices you can configure the DHCPv6 relay agent.

DHCP relay agent for IPv6

A client locates a DHCPv6 server using a reserved, link-scoped multicast address. Direct communication between the client and server requires that they are attached by the same link. In some situations where ease-of-management, economy, and scalability are concerns, you can allow a DHCPv6 client to send a message to a DHCPv6 server using a DHCPv6 relay agent.

A DHCPv6 relay agent, which may reside on the client link, but is transparent to the client, relays messages between the client and the server. Multiple DHCPv6 relay agents can exist between the client and server. DHCPv6 relay agents can also receive relay-forward messages from other relay agents; these messages are forwarded to the DHCPv6 server specified as the destination.

When the relay agent receives a message, it creates a new relay-forward message, inserts the original DHCPv6 message, and sends the relay-forward message as the DHCPv6 server.

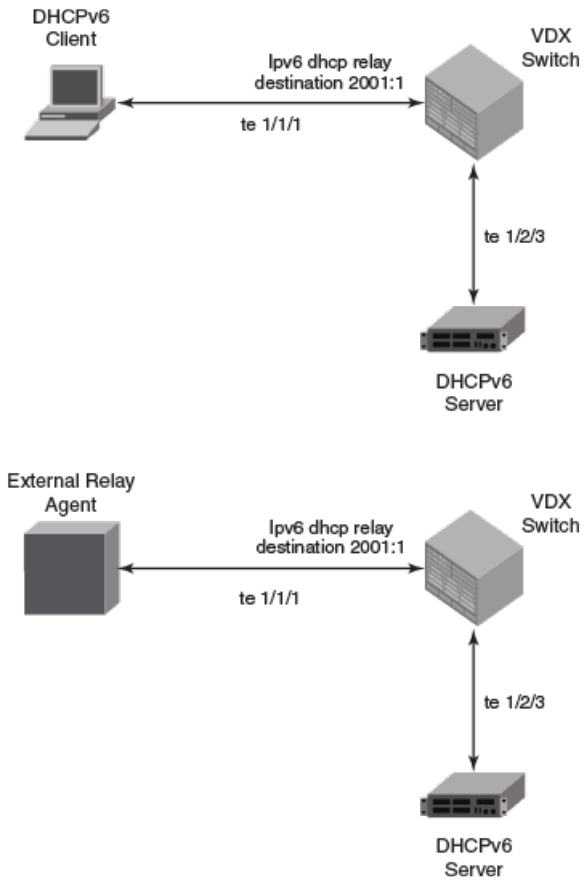
DHCPv6 relay agent

A DHCPv6 relay agent, which may reside on the client's link, is used to relay messages between the client and the server when they are not on the same IPv6 link.

The DHCPv6 relay agent operation is transparent to the client. A DHCPv6 client locates a DHCPv6 server using a reserved, link-scoped multicast address. For direct communication between the DHCPv6 client and the DHCPv6 server, both the client and the server must be attached to the same link. However, in some situations where ease of management, economy, or scalability is a concern, it is desirable to allow a DHCPv6 client to send a message to a DHCPv6 server that is not connected to the same link.

The first figure depicts the relay agent functionality when the relay agent receives DHCP packets from a client on the same link. The second figure depicts the relay agent functionality when the relay agent receives DHCP packets from another relay agent on the same link.

FIGURE 18 DHCPv6 relay functionality



DHCPv6 multicast addresses and UDP ports

The relay agent uses specific multicast addresses and UDP ports for the DHCPv6 functionality.

Multicast addresses

All_DHCP_Relay_Agents_and_Servers (FF02::1:2) is a link-scoped multicast address used by the client to communicate with neighboring (for example, on-link) relay agents and servers. All servers and relay agents are members of this multicast group.

All_DHCP_Servers (FF05::1:3) is a site-scoped multicast address used by the relay agent to communicate with servers, either because the relay agent wants to send messages to all servers or because it does not know the unicast addresses of the servers. To use this address a relay agent must have an address of sufficient scope to be reachable by the servers. All servers within the site are members of this multicast group.

UDP ports

The relay agent listens on UDP ports 546 and 547 for packets sent by clients and servers. The relayed packets use the source port 547.

DHCPv6 address assignment

The DHCPv6 relay agent informs hosts to use one of the following address assignment methods.

Basic DHCPv6 relay assignment

The DHCPv6 Relay agent relays the DHCP messages from clients and other relay agents to a list of destination addresses, which includes unicast IPv6 addresses, the All_DHCP_Servers multicast address, or other user-defined IPv6 multicast group address, on the same VRF as the interface on which the client resides or on a different VRF.

Stateful DHCPv6

DHCPv6 can be used separately, or in addition to stateless address auto-configuration (SLAAC) to obtain configuration parameters. With stateful DHCPv6, the router can signal the attached clients to use DHCPv6 to acquire an IPv6 address by setting the "O" and "M" bits in the Router Advertisement (RA) message.

The following commands set these bits to support stateful DHCPv6:

ipv6 nd managed-config-flag - When set, clients use DHCPv6 for address configuration.

ipv6 nd other-config-flag - When set, hosts use DHCPv6 to obtain other (non-address) information when set. Relay forwarding supports these flags.

Stateless DHCPv6

In stateless DHCPv6, a device uses stateless address auto-configuration (SLAAC) to assign one or more IPv6 addresses to an interface, while it utilizes DHCPv6 to receive additional parameters, such as DNS or NTP server addresses, that may not be available through SLAAC. This is accomplished by setting the "O" bit in the relay agent, using the following CLI:

ipv6 nd other-config-flag - When set, hosts use DHCPv6 to obtain other (non-address) information.

DHCPv6 prefix delegation

The DHCPv6 relay agent relays all the DHCPv6 messages (which may or may not contain the DHCPv6 options) that are to be relayed across the DHCPv6 client and the server. The relay agent will not access or extract the information present in the prefix delegation option.

Relay chaining

DHCPv6 messages can be relayed through multiple relay agents. The Relay-Reply message from the server will be relayed back to the client in the same path it took to reach the server.

Relay-message option

The DHCPV6 relay agent includes the relay message option in all the RELAY-FORW messages.

Remote-ID option

The DHCPv6 relay agent supports the Remote-ID option (option code 37). No user configuration is necessary for this method. The DHCPv6 unique identifier (DUID) of relay agent is used as the remote ID.

Interface-ID option

The DHCPv6 relay agent supports the Interface-ID option to identify the interface on which the client message was received. No user configuration is necessary for this method.

DHCPv6 message format

The DHCPv6 message format varies from the DHCPv4 message format. This section describes the events that occur when a DHCPv6 client sends a Solicit message to locate DHCPv6 servers.

- Solicit (1) - A DHCPv6 client sends a Solicit message to locate DHCPv6 servers.
- Advertise (2) - A server sends an Advertise message to indicate that it is available for DHCP service, in response to a Solicit message received from a client.
- Request (3) - A client sends a Request message to request configuration parameters, including IP addresses or delegated prefixes, from a specific server.
- Confirm (4) - A client sends a Confirm message to any available server to determine whether the addresses it was assigned are still appropriate to the link to which the client is connected. This could happen when the client detects either a link-layer connectivity change or if it is powered on and one or more leases are still valid. The confirm message confirms whether the client is still on the same link or whether if it has been moved. The actual lease(s) are not validated; just the prefix portion of the addresses or delegated prefixes.
- Renew (5) - A client sends a Confirm message to any available server to determine whether the addresses it was assigned are still appropriate to the link to which the client is connected. This could happen when the client detects either a link-layer connectivity change or if it is powered on and one or more leases are still valid. The confirm message confirms whether the client is still on the same link or whether it has been moved. The actual lease(s) are not validated; just the prefix portion of the addresses or delegated prefixes.
- Rebind (6) - A client sends a Rebind message to any available server to extend the lifetimes on the addresses assigned to the client and to update other configuration parameters; this message is sent after a client receives no response to a Renew message.
- Reply (7) - A server sends a Reply message containing assigned addresses and configuration parameters in response to a Solicit, Request, Renew, Rebind message received from a client. A server sends a Reply message containing configuration parameters in response to an Information-request message. A server sends a Reply message in response to a Confirm message confirming or denying that the addresses assigned to the client are appropriate to the link to which the client is connected. A server sends a Reply message to acknowledge receipt of a Release or Decline message.
- Release (8) - A client sends a Release message to the server that assigned addresses to the client to indicate that the client will no longer use one or more of the assigned addresses.
- Decline (9) - A client sends a Decline message to a server to indicate that the client has determined that one or more addresses assigned by the server are already in use on the link to which the client is connected.
- Reconfigure (10) - A server sends a Reconfigure message to a client to inform the client that the server has new or updated configuration parameters, and that the client needs to initiate a Renew/Reply or Information-request/Reply transaction with the server in order to receive the updated information.

- Information-request (11) - A client sends an Information-request message to a server to request configuration parameters without the assignment of any IP addresses to the client.
- Relay-forward (12) - A relay agent sends a Relay-forward message to relay messages to servers, either directly or through another relay agent. The received message, either a client message or a Relay-forward message from another relay agent, is encapsulated in an option in the Relay-forward message.
- Relay-reply - A server sends a Relay-reply message to a relay agent containing a message that the relay agent delivers to a client. The Relay-reply message may be relayed by other relay agents for delivery to the destination.

NOTE
The **show ipv6 dhcp relay statistics** command does not display the show packet count in the statistics for DHCPv6 Advertise and Re-configure messages, as it is sent from the DHCPv6 server to DHCPv6 client directly, not through the relay server.

The table lists the DHCPv4 and DHCPv6 message types:

TABLE 20 DHCPv4 message versus DHCPv6 message

DHCPv6 message type	DHCPv4 message type
Solicit (1)	DHCPDISCOVER
Advertise (2)	DHCPOFFER
Request (3), Renew (5), Rebind (6)	DHCPREQUEST
Reply (7)	DHCPPACK/DHCPNAK
Release (8)	DHCPRELEASE
Information-Request (11)	DHCPINFORM
Decline (9)	DHCPDECLINE
Confirm (4)	None
Reconfigure (10)	DHCPFORCERENEW
Relay-Forward(12), Relay-Reply (13)	None

DHCPv6 relay provisioning

Use the scalability numbers in the following table to create and apply the IPv6 DHCP relay configuration on Layer 3 interfaces.

The table lists the scalability numbers for VDX 8770 and VDX 6740:

TABLE 21 Scalability numbers

DHCP configuration	VDX 8770 (per device)	VDX 6740 (per device)
Total Number of DHCPv4 server addresses (**ip dhcp relay address**)	4000	2000
DHCPv4 server addresses per Interface (**ip dhcp relay address**)	16	16
Total number of DHCPv6 server addresses (**ipv6 dhcp relay address**)	4000	2000
DHCPv6 server addresses per interface (**ipv6 dhcp relay address**)	16	16
Number of inter-VRF client-server configurations for v4 or v6 relay (maximum size of the HSL database for each address family)	128	128
Maximum number of client packets (burst rate-limit) - client bindings per second	256 packets per second	1000 packets per second

Configuring IPv6 DHCP relay

Configure the IPv6 DHCP relay agent on any Layer 3 (L3) interface using the IP address of the DHCP server where client requests are to be forwarded.

Layer 3 interfaces can be a virtual Ethernet (VE), or physical interfaces.

You can configure the IPv6 DHCP relay agent using the **ipv6 dhcp relay address** command followed by the IP address of the DHCP server. Use the **use-vrf** *vrf-name* parameter if the DHCP server and client interface are on different Virtual Forwarding and Routing (VRF) instances. You can configure up to 16 relay destination addresses per interface.

L3 interfaces can be a virtual Ethernet (VE) or a physical 1, 10, or 40 GbE interface.

The following are the considerations when configuring IPv6 DHCP Relay:

- If the relay address is a link local address or a multicast address, an outgoing interface must be configured for IPv6 relay to function.
- In instances where the server address is relayed to a different VRF compared to client connected interface VRF, in addition to the relay address you must also specify the user-vrf, otherwise IPv6 relay may not function correctly.
- IPv6 route leaking is required for IPv6 reachability.
- Use the following commands in addition to the **ipv6 dhcp relay address** command on the layer 3 interface level so that client goes to the BOUND state.

  For example, a windows device used as a DHCPv6 client looks for these M and O flags in their received router advertisements before initialization.

  device (config)# int te 47/8/10

  device (conf-if-te-47/8/10)# ipv6 nd managed-config-flag (When set, clients use the DHCPv6 protocol for address configuration)

  device (conf-if-te-47/8/10)# ipv6 nd other-config-flag (When set, hosts use DHCPv6 to obtain 'other' non-address information)

The following scalability numbers apply to IPv6 DHCP relay:

- The total number of DHCPv6 server addresses is 4000.
- The total number of DHCPv6 server addresses per interface is 16.
- The total number of inter-vrf Client-Server configurations for DHCPv4 and v6 relay is 128.

Perform the following steps to configure an IPv6 DHCP Relay:

1. In privileged EXEC mode, issue the **configure terminal** command to enter the global configuration mode.

   ```
   device# configure
   ```

2. Enter the **interface** command followed by the interface ID to enter the interface configuration mode where you want to configure the IP DHCP Relay.

   ```
   device# interface ethernet 1/17
   device# interface ethernet 0/4
   device(config)# int ve 100
   ```

3. Enter the IP address of the DHCP server followed by the interface number.

   ```
   device(config-if-eth-1/17)# ipv6 dhcp relay address 2000::10
   device(config-if-eth-1/17# ipv6 dhcp relay address fe80::1016 interface ethernet 2/67
   ```

4. Enter the IP address of the DHCP server followed by the interface number.

```
device(conf-if-eth-0/4)# ipv6 dhcp relay address 2000::10
device(conf-if-eth-0/4)# ipv6 dhcp relay address fe80::1016 interface ethernet 0/6
```

5. For VE interfaces, enter the **ipv6 dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command where *ip-addr* is the IP address of the DHCP server. Use the **use-vrf** *vrf-name* option if the DHCP server is on a different VRF instance than the interface where the client is connected.

```
(config)# int ve 100
device(config-Ve-100)# ipv6 dhcp relay address 2001::1122:AABB:CCDD:3344 use-vrf blue
```

6. Enter the **ipv6 dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command where *ip-addr* is the IP address of the DHCP server. Use the **use-vrf** *vrf-name* option if the DHCP server is on a different VRF instance than the interface where the client is connected.

```
(config)# int ve 100
device(config-Ve-100)# ipv6 dhcp relay address 2001::1122:AABB:CCDD:3344 use-vrf blue
```

7. To remove the IP DHCP Relay address enter the **no ipv6 dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command.

The following example specifies the IP DHCP relay address on VE 100 and a ten gigabit ethernet interface.

```
device(config)# int ve 100
device(config-Ve-100)# ipv6 dhcp relay address 2001::1122:AABB:CCDD:3344 use-vrf blue
device(config)# int TenGiga 2/3/1
switch(config-if)# ipv6 dhcp relay address fe80::224:38ff:febb:e3c0  interface tengigabitethernet 2/5
```

Displaying DHCPv6 relay addresses on specific devices

Use the show ipv6 dhcp relay address rbridge-id command to display all IP DHCP Relay addresses configured on specific switches (specified by RBridge IDs) in a logical chassis cluster. You can use the **all** parameter to display configured addresses on all RBridge IDs in a cluster.

Use the **show ipv6 dhcp relay address** command to display all IP DHCP Relay addresses configured on specific devices.

1. Access a device where an IP DHCPv6 Relay has been configured.
2. In privileged EXEC mode, execute the **show ipv6 dhcp relay address rbridge-id** *rbridge-id* command.
3. In privileged EXEC mode, execute the **show ipv6 dhcp relay address** command.

The following example displays addresses configured on the interfaces of a device.

```
device# show ipv6 dhcp relay address

DHCPv6 unique identifier(DUID): 0102f8f10027f8d43dfb

Interface           Relay Address                           VRF Name            Outgoing Interface
---------           -------------                           --------            ------------------
eth 1/17             2002::10                                default-vrf
eth 1/17             2000::10                                default-vrf          eth 0/9
eth 1/17             2019::10                                test                 ve 102
ve 94                2011::10                                default-vrf
```

The following example displays addresses configured on the interfaces of a device.

```
device# show ipv6 dhcp relay address

DHCPv6 unique identifier(DUID): 0102f8f10027f8d43dfb

Interface           Relay Address                           VRF Name            Outgoing Interface
---------           -------------                           --------            ------------------
eth 0/17             2002::10                                default-vrf
eth 0/17             2000::10                                default-vrf         eth 0/9
eth 0/17             2019::10                                test                ve 102
ve 94                2011::10                                default-vrf
```

The following is an example of displaying addresses configured on interfaces of a local switch. Notice that the RBridge ID is not needed in the command.

```
device# show ipv6 dhcp relay address

              RBridge Id:  2
              ----------------
DHCPv6 unique identifier(DUID): 0102111f70027f8cad50d

Interface     Relay Address                           VRF Name                Outgoing Interface
---------     -----------                             --------                ------------------

Te 2/0/35      4001::101                               blue                   Te 2/0/34
Te 2/0/35       fe80::8                                blue                   Ve 100
Ve 200    5001:1234:1234:2101:1234:1234:3103:1234   default-vrf               Te 2/0/2
```

NOTE

You can specify a list of RBridge IDs separated by commas, or a range separated by a dash (for example, 1-2). No spaces are allowed in the range string. The range does not need to be contiguous (for example, 1-2,5). You can also specify **all** for all RBridge IDs in the logical chassis cluster.

The following is an example of displaying addresses configured on interfaces on a specific RBridge.

```
device# show ipv6 dhcp relay address rbridge-id 2

               RBridge Id: 2
               ----------------
DHCPv6 unique identifier(DUID): 0102111f70027f8cad50d

Interface      Relay Address                              VRF Name        Outgoing Interface
-------------  -------------                              --------        ------------------
Te 2/0/35      4001::101                                   blue            Te 2/0/34

Te 2/0/35      fe80::8                                     blue            Ve 100

Ve 200        5001:1234:1234:2101:1234:1234:3103:1234     default-vrf     Te 2/0/2
```

The following is an example of displaying addresses configured on interfaces on all RBridge IDs in a logical chassis cluster.

```
device# show ipv6 dhcp relay address rbridge-id all

               RBridge Id: 2
               -------------
DHCPv6 unique identifier(DUID): 0102111f70027f8cad50d

Interface      Relay Address                          VRF Name          Outgoing Interface
----------     -------------                          ---------         ------------------
Te 2/0/35       4001::101                             blue               Te 2/0/34

Te 2/0/35       fe80::8                               blue               Ve 100
Ve 200     5001:1234:1234:2101:1234:1234:3103:1234   default-vrf        Te 2/0/2

               RBridge Id: 3
               -------------
DHCPv6 unique identifier(DUID): 0102a1b00027f8cb13ab

Interface      Relay Address                                      VRF Name
----------     -------------                                     ---------
Ve 300         5001:1234:1234:2101:1234:1234:3103:1234           default-vrf
```

Displaying DHCPv6 relay addresses for an interface

You can display IPv6 DHCP Relay addresses configured on a specific interfaces of the device.

To display the IPv6 DHCP Relay addresses configured for an interface, use the **show ipv6 dhcp relay address interface** command followed by the specific interface to display IP DHCP Relay addresses configured on a specific interface.

1. Access a device where an IP DHCP Relay has been configured on an interface.
2. In privileged EXEC mode, execute the **show ip dhcp relay address interface** command followed by the interface ID.

The following example displays the IPv6 DHCP Relay addresses configured on interface 1/40.

```
device# show ipv6 dhcp relay address interface ethernet 1/40

DHCPv6 unique identifier(DUID): 01021025768ef804e005

Interface           Relay Address                          VRF Name            Outgoing Interface
---------           -------------                          --------            ------------------
Eth 1/40             2019::10                               default-vrf
```

The following example displays the IPv6 DHCP Relay addresses configured on interface 0/4.

```
device# show ipv6 dhcp relay address interface ethernet 0/4

DHCPv6 unique identifier(DUID): 01021025768ef804e005

Interface           Relay Address                          VRF Name            Outgoing Interface
---------           -------------                          --------            ------------------
Eth 0/4             2019::10                               default-vrf
```

The following example displays the IPv6 DHCP Relay addresses configured on interface ve 109.

```
device# show ipv6 dhcp relay address interface ve 109

DHCPv6 unique identifier(DUID): 01021025768ef804e005

Interface           Relay Address                          VRF Name            Outgoing Interface
---------           -------------                          --------            ------------------
Ve 109               2020::20                               default-vrf
```

The following is an example for a 10GbE interface of a local switch.

```
device# show ipv6 dhcp relay address interface Te 3/0/21
                                  Rbridge Id:    3
                                   ----------------

DHCPv6 unique identifier(DUID): 0102111f70027f8cad50d

Interface               Relay Address                     VRF Name           Outgoing Interface
---------              -------------                 -------------          -----------------------
Te 3/0/21             4001::101                        default-vrf
Te 3/0/21               fe80::8                            blue                          ve 100
```

The following is an example for a Virtual Ethernet interface on a specific switch (RBridge ID 1).

```
device# show ipv6 dhcp rel add int ve 300 rbridge-id 1

              RBridge Id:    1
              ----------------
DHCPv6 unique identifier(DUID): 0102111f70027f8cad50d

Interface     Relay Address       VRF Name      Outgoing Interface
---------     -------------       --------      -------------------
Ve 300        fe80::8           default-vrf
```

The following is an example for displaying addresses on for a specific Virtual Ethernet interface on a range of switches (specified by RBridge IDs) in a logical chassis cluster.

> **NOTE**
> You can specify a list of RBridge IDs separated by commas, or a range separated by a dash (for example, 1-2). No spaces are allowed in the range string. The range does not need to be contiguous (for example, 1-2,5). You can also specify “all” for all RBridge IDs in the logical chassis cluster.

```
device# show ipv6 dhcp rel add int ve
300 rbridge-id 1,3

             RBridge Id:    1
             ----------------
DHCPv6 unique identifier(DUID): 0102111f70027f8cad50d

Interface    Relay Address        VRF Name             Outgoing Interface
---------    -------------        --------             ------------------
Ve 300       4001::101             default-vrf
Ve 300       4001::102             default-vrf

             RBridge Id: 3
             -------------
DHCPv6 unique identifier(DUID): 0102a1b00027f8cb13ab

Interface    Relay Address                        VRF Name         Outgoing Interface
---------    -------------                         --------        ------------------
Ve 300       2001::1122:AABB:CCDD:3344              blue
Ve 300       fe80::224:38ff:febb:e3c0               green
Ve 300       2001::1122:AABB:CCDD:3355             default-vrf
```

Displaying IPv6 DHCP relay statistics

Display information about the DHCP Relay function, such as the DHCP Server IP address configured on the device and the number of various DHCP packets received by the interface configured for IP DHCP Relay.

Use the **show ipv6 dhcp relay statistics** command to display the following information about the IPv6 DHCP Relay function:

- Number of SOLICIT, REQUEST, CONFIRM, RENEW, REBIND, RELEASE, DECLINE, INFORMATION-REQUEST, RELAY-FORWARD, RELAY-REPLY packets received.

- Number of RELAY-FORWARD and REPLY packets sent and packets dropped.

  NOTE
  You can specify a list of RBridge IDs separated by commas, or a range separated by a dash (for example, 1-2). No spaces are allowed in the range string. The range does not need to be contiguous (for example, 1-2,5). You can also specify "all" for all RBridge IDs in the logical chassis cluster.

1. Access a device where an IPv6 DHCP Relay has been configured on an interface.
2. In privileged EXEC mode, enter **show ipv6 dhcp relay statistics**. Optionally, you can specify specific RBridge IDs if you only want to view the statistics for those RBridges.
3. In privileged EXEC mode, enter **show ipv6 dhcp relay statistics**.

The following is an example of displaying statistics on a local switch.

```
device# show ipv6 dhcp relay statistics
                             Rbridge Id: 10
Packets dropped : 0
  Error : 0
Packets received : 60
  SOLICIT : 6
  REQUEST : 6
  CONFIRM : 0
  RENEW : 2
  REBIND : 0
  RELEASE : 6
  DECLINE : 0
  INFORMATION-REQUEST : 0
  RELAY-FORWARD : 0
  RELAY-REPLY : 40
Packets sent : 60
  RELAY-FORWARD : 20
  REPLY : 40
```

The following is an example of displaying statistics on RBridge 10.

```
device# show ipv6 dhcp relay statistics rbridge-id 10

                             Rbridge Id: 10
Packets dropped : 0
  Error : 0
Packets received : 60
  SOLICIT : 6
  REQUEST : 6
  CONFIRM : 0
  RENEW : 2
  REBIND : 0
  RELEASE : 6
  DECLINE : 0
  INFORMATION-REQUEST : 0
  RELAY-FORWARD : 0
  RELAY-REPLY : 40
Packets sent : 60
  RELAY-FORWARD : 20
  REPLY : 40
```

The following example displays DHCP relay statistics on the device.

```
device# show ipv6 dhcp relay statistics

DHCPv6 Relay Statistics
------------------------
Packets dropped : 0
  Error : 0
Packets received : 60
  SOLICIT : 6
  REQUEST : 6
  CONFIRM : 0
  RENEW : 2
  REBIND : 0
  RELEASE : 6
  DECLINE : 0
  INFORMATION-REQUEST : 0
  RELAY-FORWARD : 0
  RELAY-REPLY : 40
Packets sent : 60
  RELAY-FORWARD : 20
  REPLY : 40
```

Clearing IP DHCPv6 relay statistics

Use the **clear ip dhcpv6 relay statistics** command to clear all IPv6 DHCP Relay statistics for specific relay IP addresses, for addresses on specific RBridge IDs, or all addresses for RBridge IDs in a logical chassis cluster.

Use the **clear ip dhcpv6 relay statistics** command to clear all IPv6 DHCP Relay statistics globally.

1. Access a device where an IPv6 DHCP Relay has been configured on an interface.
2. In privileged EXEC mode, issue the **clear ipv6 dhcp relay statistics ip-address** *ip-address* **rbridge-id** *rbridge-id*.
3. In privileged EXEC mode, issue the **clear ipv6 dhcp relay statistics ip-address** *ip-address* command.

The following command clears the IPv6 DHCP Relay address statistics globally.

```
device# clear ipv6 dhcp relay statistics
```

The following command clears statistics for IPv6 DHCP Relay address 10.1.0.1 configured on RBridges 1, 3, and 5.

```
clear ipv6 dhcp relay statistics ip-address 10.1.0.1 rbridge-id 1,3,5
```

BFD

- Bidirectional Forwarding Detection overview....169
- General BFD considerations and limitations....170
- BFD on Network OS hardware platforms....171
- BFD for Layer 3 protocols....173
- BFD considerations and limitations for Layer 3 protocols....175
- BFD for multihop ECMP....176
- BFD for Layer 3 protocols on virtual Ethernet interfaces....176
- BFD for Layer 3 protocols on vLAGs....177
- BFD for L3 protocols on port channels....177
- Configuring BFD on an interface....178
- Configuring BFD on an interface....178
- Disabling BFD on an interface....179
- BFD for BGP....180
- BFD for OSPF....190
- BFD for IS-IS....196
- BFD for VXLAN extension tunnels....197
- Configuring BFD on a VXLAN extension tunnel....201
- BFD for NSX tunnels....202
- BFD for static routes....202
- BFD for RSVP-TE LSP....208
- Displaying BFD information....214
- Displaying BFD information....216
- Displaying BFD information....220
- Displaying BFD information....222
- Displaying BFD information....225

Bidirectional Forwarding Detection overview

Bidirectional Forwarding Detection (BFD) is a unified detection mechanism used to rapidly detect link faults. BFD improves network performance by providing fast forwarding path failure detection times.

BFD provides rapid detection of the failure of a forwarding path by checking that the next-hop device is alive. When BFD is not enabled, it can take time to detect that a neighboring device is not operational. This causes packet loss due to incorrect routing information at a level unacceptable for real-time applications such as VOIP and video over IP.

Using BFD, you can detect a forwarding path failure in 300 milliseconds or less, depending on your configuration.

Using BFD, you can detect a forwarding path failure in 150 milliseconds.

A BFD session is automatically established when a neighbor is discovered for a protocol, provided that BFD is enabled on the interface on which the neighbor is detected and BFD is also enabled for the protocol at the interface level or globally. Once a session is established, each device transmits control messages at a high rate of speed that is negotiated by the devices during the session setup. To provide a detection time of 150 milliseconds, it is necessary to process 20 messages per second of about 70 to 100 bytes each per session. A similar number of messages also need to be transmitted out per session. Once a session is established, that same message is continuously transmitted at the negotiated rate and a check is made that the expected control message is received at the agreed frequency from the neighbor. If the agreed upon messages are not received from the neighbor within a negotiated timeout period, the neighbor is considered to be down.

For CER 2000 Series and CES 2000 Series devices, there are 20 Bidirectional Sessions per LP and 40 Bidirectional sessions system-wide.

BFD can provide failure detection on any kind of path between systems, including direct physical links, multihop routed paths, and tunnels. Multiple BFD sessions can be established between the same pair of systems when multiple paths between them are present in at least one direction, even if a lesser number of paths are available in the other direction.

> **NOTE**
> BFD session establishment on an interface does not start until 180 seconds after the interface comes up. The reason for this delay is to ensure that the link is not effected by unstable link conditions which could cause BFD to flap. This delay time is not user configurable.

> **NOTE**
> BFD session establishment on an interface does not start until 5 seconds after the interface comes up. The reason for this delay is to ensure that the link is not affected by unstable link conditions which could cause BFD to flap. This delay time is not user configurable.

The BFD Control Message is an UDP message with destination port 3784.

For a single-hop session, the BFD Control Message is a UDP message with destination port 3784. For a multihop session, the BFD Control Message is a UDP message with destination port 4784 sent over IPv4 or IPv6, depending on which data forwarding path failure BFD is trying to detect.

> **NOTE**
> The source port for BFD control packets is in the range 49152 through 65535. The source port number is unique among all BFD sessions on the system.

> **NOTE**
> For single-hop sessions, all BFD control packets are sent with a time to live (TTL) or hop limit value of 255. All received BFD control packets are discarded if the received TTL or hop limit is not equal to 255.

> **NOTE**
> BFD supports multi-slot LAGs in cases where all BFD packets are transmitted only on a single path which does not change unless the LAG active membership changes. BFD is not be supported on multi-slot LAGs where per-packet switching is used such that the path taken by the BFD packets will vary per packet.

> **NOTE**
> When BFD is configured with stringent values of 100/300 msec, BFD may flap when learning a large number of routes.

> **NOTE**
> BFD sessions configured with lower timer values may exhibit flaps when configured alongside MACSec on same line card. This issue is a known limitation. However, the BFD sessions are stable with 250ms*3 timer value or more in such scenarios.

General BFD considerations and limitations

There are a number of general points to consider when configuring BFD:

- BFD is not supported on Layer 3 port channels.
- BFD is supported on Layer 2 port channels used by SVI sessions. Refer to the *Extreme Network OS Layer 2 Switching Configuration Guide* for more information on Layer 2 port channels. For trunk ports, active BFD sessions are started on the line card (LC) with the primary port. Inactive BFD sessions are started on all other LCs with secondary member ports. The BFD

state machine is replicated on all LCs. Inactive sessions on other LCs receive control packets that arrive from alternate paths. Session timeouts are triggered if control packets are not received on any path.

- BFD protocol version 1 is supported. BFD version 0 is not supported. BFD version 1 and BFD version 0 are not compatible.
- BFD single-hop sessions always use the primary IP address as the source address. Secondary IP addresses cannot be used as source addresses on single-hop sessions.
- · 40 BFD sessions are supported per line card

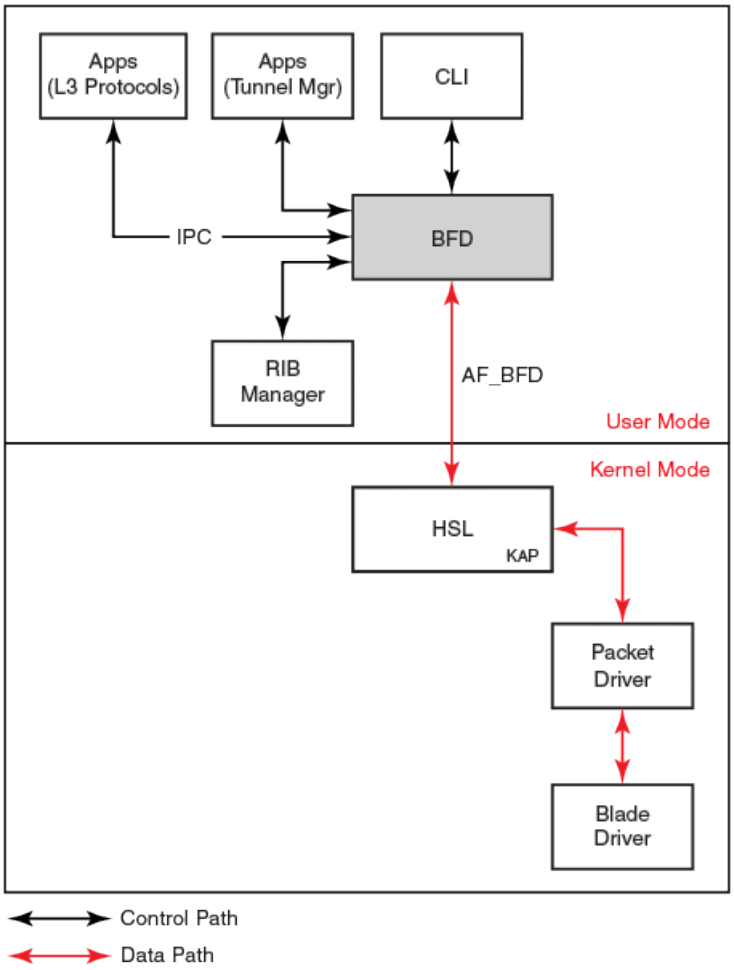
Refer to BFD considerations and limitations for Layer 3 protocols on page 175 and the *BFD considerations and limitations for static routes* section of the "IP Route Policy" chapter for more information on BFD considerations and limitations.

BFD on Network OS hardware platforms

A single instance of BFD runs on stackable switch platforms, and BFD sessions cannot be offloaded. On chassis-based platforms support for offloading BFD sessions to line cards is provided, helping to achieve scalability.

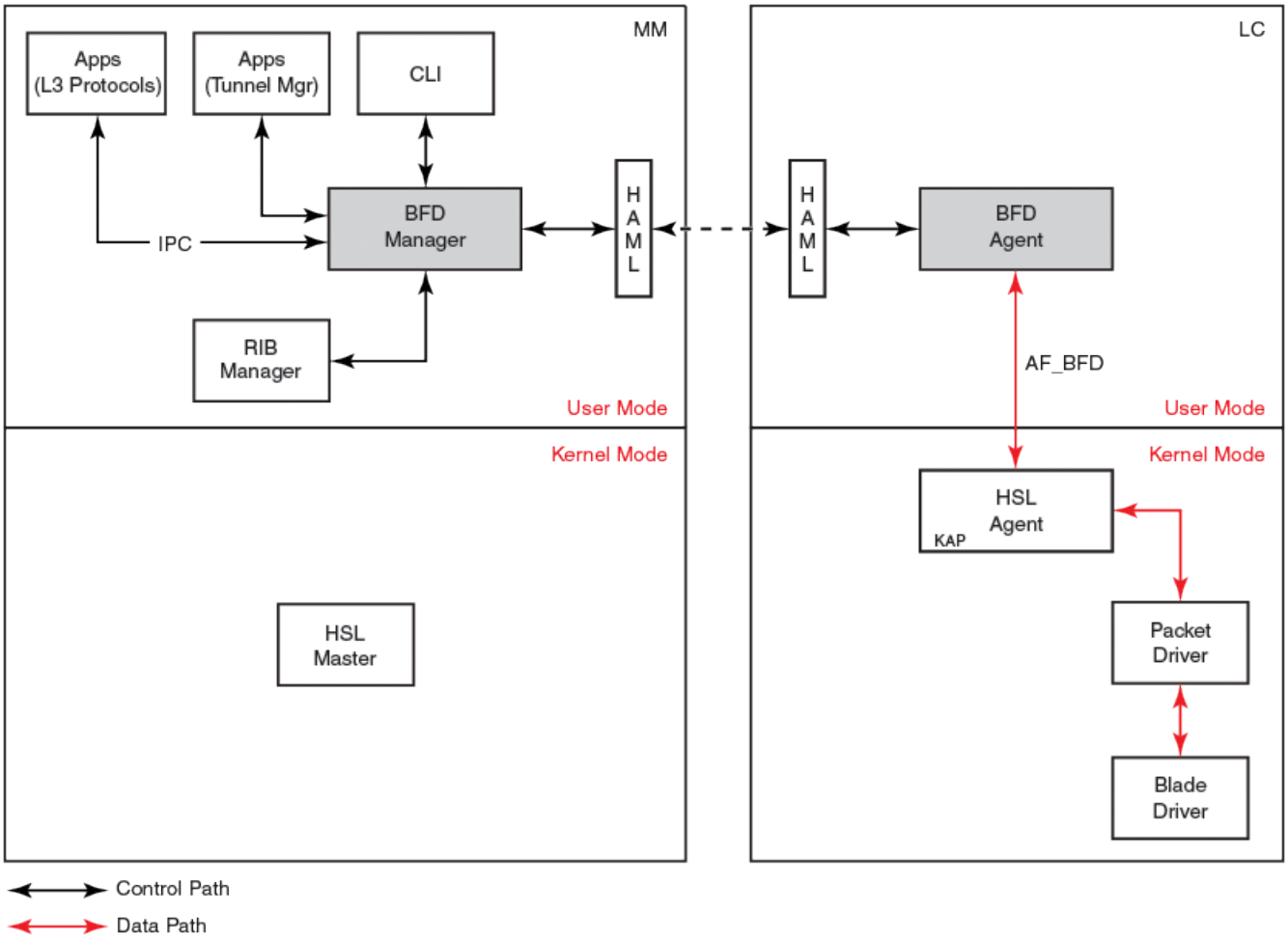
The figure below shows BFD running on a stackable switch platform, where BFD sessions are not offloaded. BFD is responsible for providing the forwarding path failure detection service to a routing protocol.

FIGURE 19 BFD module on a stackable switch



The figure below shows BFD running on a chassis-based platform, where BFD sessions are offloaded to a LC. BFD is responsible for actually detecting the failure of the forwarding path.

FIGURE 20 BFD module on a chassis-based platform



BFD for Layer 3 protocols

BFD can be used by Layer 3 protocols for rapid failure detection in the forwarding path between two adjacent routers, including the interfaces, data links, and forwarding planes.

BFD can be configured for use with the following protocols:

- OSPFv2
- OSPFv3
- IS-IS
- BGP4
- BGP4+

BFD must be enabled at both the interface and routing protocol levels. BFD asynchronous mode, which depends on the sending of BFD control packets between two systems to activate and maintain BFD neighbor sessions between routers, is supported. Therefore, in order for a BFD session to be created, BFD must be configured on both BFD peers.

Once BFD is enabled on the interfaces and at the router level for the appropriate routing protocols, a BFD session is created. BFD timers are then negotiated, and the BFD peers begin to send BFD control packets to each other at the negotiated interval.

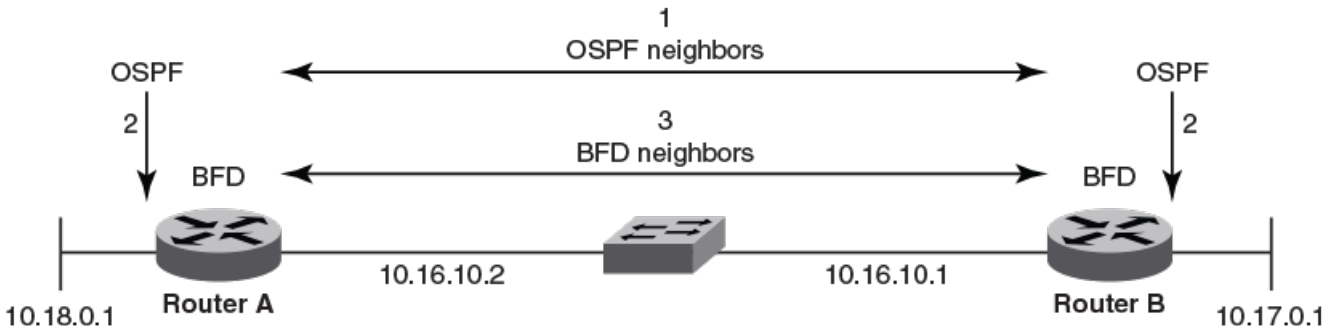
BFD provides a single point of forwarding path monitoring. This means that when more than one Layer 3 application wants to monitor a single host, BFD runs a single session for that host and provides the status to multiple applications, instead of multiple applications running individual sessions to the host.

By sending rapid failure detection notices to the routing protocols in the local device to initiate the routing table recalculation process, BFD contributes to greatly reducing overall network convergence time.

NOTE
BFD brings IS-IS and OSPF down with it when RSTP path-cost changes are made to the switch Alt Discarding port.

The following figure shows the establishment of a BFD session where OSPF discovers a neighbor and sends a request to BFD requesting that a BFD neighbor session be created with the OSPF neighbor router.

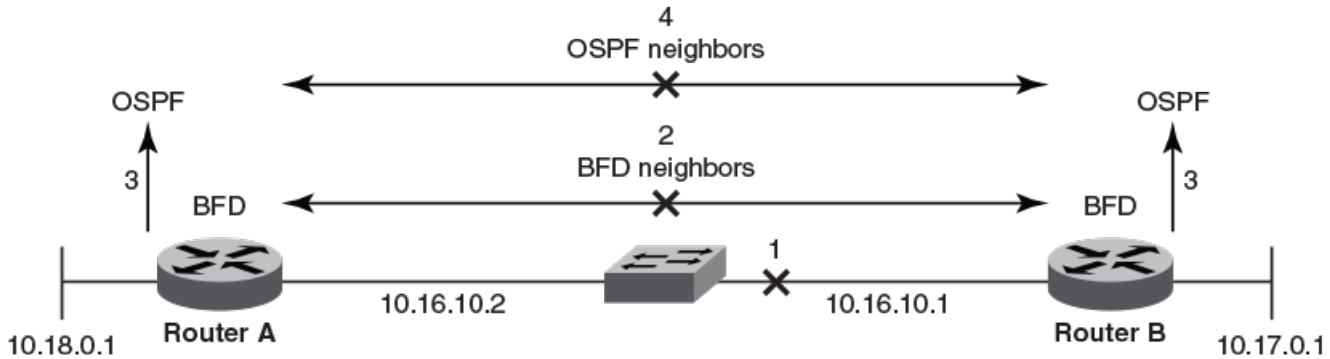
FIGURE 21 Establishing a BFD neighbor session



1. OSPF discovers a neighbor.
2. OSPF requests that the local BFD process initiate a BFD neigbor session with the OSPF neigbor router.
3. A BFD neighbor session is established with the OSPF neighbor router.

The following figure shows the termination of a BFD neighbor session after a failure occurs in the network.

FIGURE 22 Terminating a BFD neighbor session



1. A network failure occurs.
2. The BFD session with the OSPF neigbor router is torn down.
3. BFD notifies the local OSPF process that the BFD neighbor is not reachable.
4. The local OSPF process tears down the OSPF relationship and starts reconverging.

BFD considerations and limitations for Layer 3 protocols

There are a number of things to consider when configuring BFD for Layer 3 protocols:

- BFD is not supported on Layer 3 port channels. For information on BFD support on Layer 2 port channels, refer to the *Extreme Network OS Layer 2 Switching Configuration Guide*.
- BFD supports both single-hop and multihop sessions.
- For single-hop sessions, the IP address that matches the subnet of the destination IP address is always used as the source IP address of the BFD control packet. If the source IP address is changed, the session is brought down unless another corresponding address with the same subnet is available.
- For single-hop sessions, an IPv6 address that matches the prefix and scope of the destination IPv6 address is used as the source IPv6 address of the BFD control packet.
- For multihop sessions, BFD clients provide the source IP address and the Layer-3 protocol is notified of any change to the source IP address of the BFD control packet.
- Multicast or anycast address IP addresses can not be used as source IP addresses.
- BFD establishes only a single BFD session per data protocol path (IPv4 or IPv6) regardless of the number of protocols that want to utilize it.
- If a Layer 3 interface is configured, BFD is enabled by default on the interface (IPv4 or IPv6).
- Registration is global across all VRFs, even if a neighbor does not support BFD.
- Inactive sessions are created when BFD is not enabled on a remote device that contains the destination IP address. Sessions created are in the Admin Down state and are transmitted at a slower rate than configured values.
- BFD sessions can be established on both primary and secondary IP and IPv6 addresses.
- BFD sessions can be established on link-local IPv6 addresses.
- Changing the BFD parameters does not reset the current BFD session.
- When BFD notifies BGP or OSPF that a session has transitioned from up to down, the protocol does not immediately bring down the session if the holdover timer is configured. The protocol waits until the period of time specified for the holdover timer has expired. If BFD declares a session up before this period of time expires, no action is taken by the protocol.
- BFD sessions remain intact, even after a HA failover, unless there is no topology change or the BFD session went down from the remote end.
- On point-to-point links, the source address of a BFD control packet cannot be used to identify the session.
- If you unconfigure a BFD session that is in the up state, OSPF or BGP tell BFD to delete the session and set the reason as Admin Down. Upon receipt of this notification, BFD deletes the session and communicates this change to the remote BFD peer. The remote BFD neighbor keeps the session in the down state and propagates Remote Admin Down event to the routing protocols.
- BFD session establishment on an interface does not start until 180 seconds after the interface comes up. The reason for this delay is to ensure that the link is not effected by unstable link conditions which could cause BFD to flap. This delay time is not user configurable.

BFD for multihop ECMP

For equal-cost multi-path (ECMP) routing, only one BFD session over one path is maintained.

BFD registers the destination IP address with RIB Manager. RIB Manager uses ARP to resolve all next-hop addresses. BFD caches Layer-3 route reachability information and Layer-2 resolved-ARP information that is provided by RIB Manager.

When ARP ages out for a nexthop, RIB Manager automatically issues another ARP request to resolve it. RIB Manager notifies BFD of ARP-resolved information.

When a nexthop that is in use goes down, BFD transmits packets through the next ARP-resolved nexthop. When all nexthops are in an unresolved state, BFD triggers hold time (approximately 1-2 seconds) and waits for RIB Manager to resolve the ARP. When ARP is not resolved within the hold time, then a session-down notification is sent to the application. When a BFD packet receive times out on the path that is used for creating the session, then a session-down notification is sent to the protocol.

BFD for Layer 3 protocols on virtual Ethernet interfaces

BFD can be configured for Layer 3 protocols on virtual Ethernet (VE) interfaces.

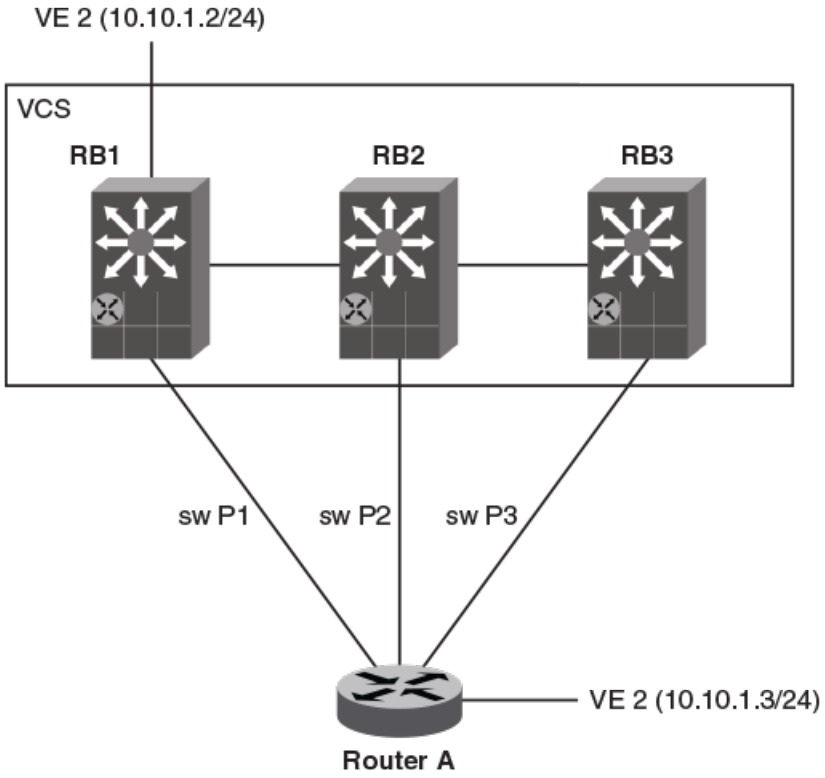
When configuring BFD for Layer 3 protocols on VE interfaces, one of the physical ports is chosen to set up the session. The physical port used for setting up the BFD session is allocated after Address Resolution Protocol (ARP) is resolved for that neighbor.

If a member of a VE port goes down and a new egress port is identified, the session is moved to another physical port. If a new egress port is not identified, BFD tries to locate a destination IPv4 or IPv6 address. If the new egress port is not learned within the BFD detection time, the session is declared as down.

BFD support for Layer 3 protocols on VEs over VCS

When a BFD session is created, the RBridge where the session is created is designated as the “master RBridge”. This master RBridge runs the full BFD state machine. In the figure below, RB1 is the master RBridge and runs the BFD session. The RBridges are connected through an Inter-Switch Link (ISL) . If the ARP-resolved egress port is in RB2, RB1 begins hardware-assisted BFD packet transmission over the ISL connected to RB2. If the switch port, swP2, goes down or RB2 gets disconnected from the VCS, RB1 begins BFD packet transmission over the next ARP resolved egress port.

FIGURE 23 BFD on a virtual Ethernet interface over VCS



BFD packets received in switch ports of Remote RBridges are redirected to the master RBridge.

BFD for Layer 3 protocols on vLAGs

BFD can be configured for Layer 3 protocols on Virtual Link Aggregation Groups (vLAGs).

When configuring BFD on a vLAG, the master RBridge begins BFD packet transmission over the local member port. If the local member port goes down, the vLAG parent virtual Ethernet (VE) interface is reachable using the port channel member port of remote RBridges. This parent VE interface remains in the Up state throughout.

If the master RBridge does not have any local member port, it begins hardware-assisted BFD packet transmission over an Inter-Switch Link (ISL) connected to a remote RBridge. If the vLAG secondary member port goes down, or the second RBridge is disconnected from the VCS, the master RBridge begins BFD packet transmission over another member port.

BFD for L3 protocols on port channels

BFD is supported on Layer-2 and Layer-3 port channels that are used by switched virtual interface (SVI) sessions.

Session timeout is triggered when a packet is not received on any path.

When BFD packet receive times out, a session-down notification is sent to the protocol.

Configuring BFD on an interface

BFD can be configured on device interfaces. Repeat the steps in this procedure for each interface on which you want to configure BFD sessions.

If a Layer 3 interface is configured, BFD is enabled by default on the interface (IPv4 or IPv6).

> NOTE
> This task does not result in BFD session creation, but configures BFD session parameters for the interface.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface** command and specify an interface.

```
device(config)# interface tengigabitethernet 1/0/1
device(config)# interface ethernet 0/4
device(config)# interface ethernet 1/4
```

3. Enter the **bfd interval** command with the **min-rx** and **multiplier** keywords to configure BFD session parameters on the interface.

```
device(config-if-te-1/0/1)# bfd interval 110 min-rx 120 multiplier 15
device(config-if-eth-0/4)# bfd interval 110 min-rx 120 multiplier 15
device(config-if-eth-1/4)# bfd interval 110 min-rx 120 multiplier 15
```

The following example configures BFD on a specific 10-gigabit Ethernet interface by setting the baseline BFD session parameters on that interface.

```
device# configure terminal
device(config)# interface tengigabitethernet 1/0/1
device(config-if-te-1/0/1)# bfd interval 110 min-rx 120 multiplier 15
```

The following example configures BFD on a specific Ethernet interface by setting the baseline BFD session parameters on that interface.

```
device# configure terminal
device(config)# interface ethernet 1/4
device(config-if-eth-1/4)# bfd interval 110 min-rx 120 multiplier 15

device# configure terminal
device(config)# interface ethernet 0/4
device(config-if-eth-0/4)# bfd interval 110 min-rx 120 multiplier 15
```

Configuring BFD on an interface

BFD can be configured on device interfaces. Repeat the steps in this procedure for each interface on which you want to configure BFD sessions.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface** command and specify an interface.

```
device(config)# interface ethernet 1/1
```

3. Enter the **bfd interval** command, specifying an interval, with the **min-rx** and **multiplier** keywords to configure BFD session parameters on the interface.

```
device(config-if-e1000-1/1)# bfd interval 110 min-rx 120 multiplier 15
```

The following example configures BFD on a specific Ethernet interface by setting the baseline BFD session parameters on that interface.

```
device# configure terminal
device(config)# interface ethernet 1/1
device(config-if-e1000-1/1)# bfd interval 110 min-rx 120 multiplier 15
```

Disabling BFD on an interface

BFD can be disabled on device interfaces. Repeat the steps in this procedure for each interface onThe following figure which you want to disable BFD sessions.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface** command and specify an interface.

```
device(config)# interface tengigabitethernet 1/0/1

device(config)# interface ethernet 0/4

device(config)# interface ethernet 1/4
```

3. Enter the **bfd shutdown** command to disable BFD on the interface.

```
device(config-if-te-1/0/1)# bfd shutdown

device(config-if-eth-0/4)# bfd shutdown

device(config-if-eth-1/4)# bfd shutdown
```

The following example disables BFD on a specific 10-gigabit Ethernet interface.

```
device# configure terminal
device(config)# interface tengigabitethernet 1/0/1
device(config-if-te-1/0/1)# bfd shutdown
```

The following example disables BFD on a specific Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 0/4
device(config-if-eth-0/4)# bfd shutdown

device# configure terminal
device(config)# interface ethernet 1/4
device(config-if-eth-1/4)# bfd shutdown
```

BFD for BGP

BFD support for BGP4 and BGP4+ can be configured so that BGP is a registered protocol with BFD and receives forwarding path detection failure messages from BFD.

BFD is supported for BGP and is disabled by default. When BFD for BGP is enabled, BFD rapidly detects faults on links between BGP peers and reports faults to BGP. BFD for BGP is supported for both for single-hop and multihop iBGP and eBGP sessions with either IPv4 or IPv6 neighbors in the default VRF and nondefault VRF instances. BFD behavior is identical for iBGP and eBGP single-hop and multihop sessions, and for IPv4 and IPv6 neighbors.

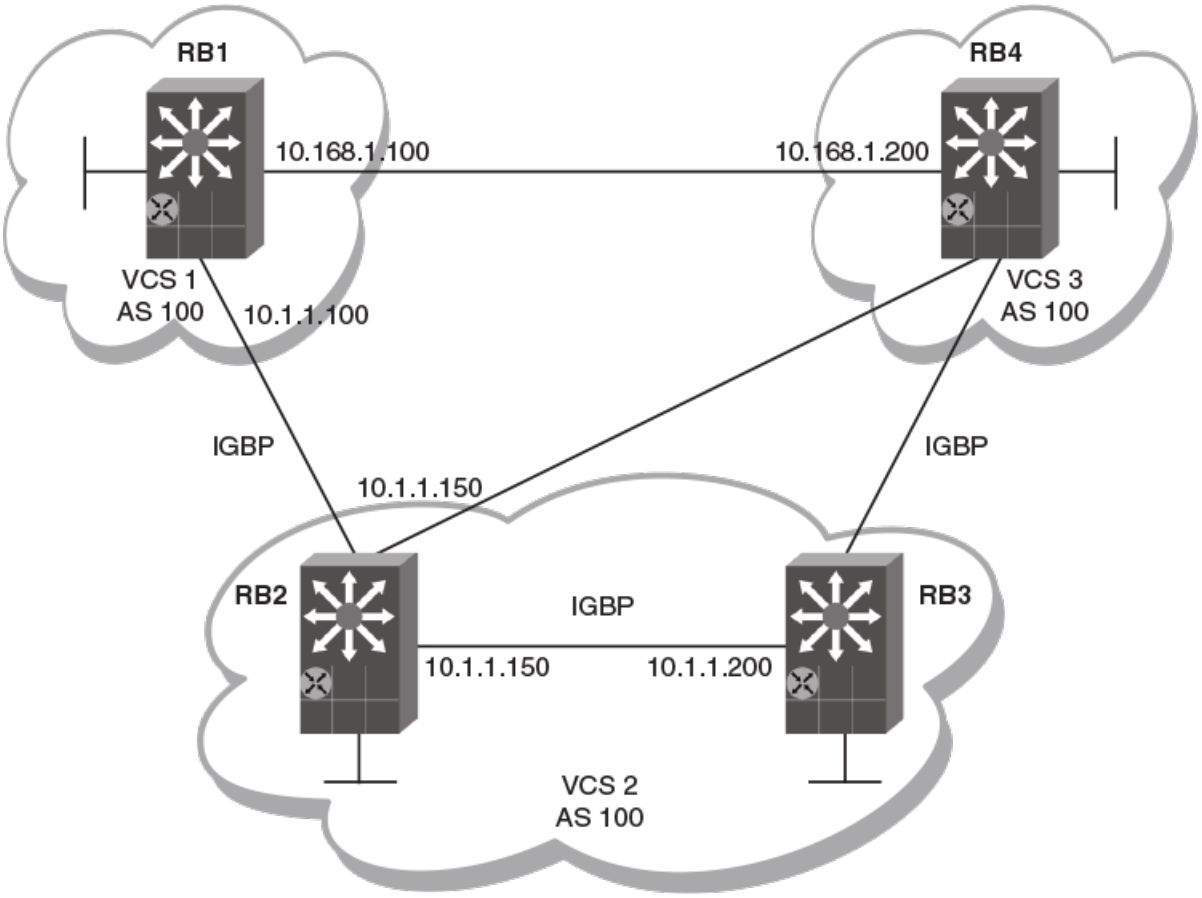
> **NOTE**
> BFD for unnumbered EMCP interfaces is supported for BGP in IP Fabric deployments. For further information, refer to the *Extreme Network OS IP Fabrics Configuration Guide*.

Consider the following when configuring BFD for BGP:

- Registration is global across all VRFs once BGP sends a registration message to BFD.
- BFD sessions for remote BGP neighbors are not triggered if BFD is not configured on these neighbors. Each neighbor can have its own transmit interval, receive internal, and detect multiplier. If the value is not configured, the configured global value is inherited.
- As soon as a BGP session enters the Established state, BGP requests that BFD start a BFD session.
- BFD sessions are maintained across a BGP graceful restart.

The figure below shows an IBGP session. This session running between RB1 and RB2 is a singlehop BFD session that tracks the remote address. The session running between RB1 and RB3 is a BFD multihop session that tracks the end points.

FIGURE 24 BFD for BGP



BFD for BGP session creation and deletion

When BGP requests that BFD start a BFD session, each session has associated BFD session parameters configured. Session parameter values are selected according to a specific hierarchy.

For multihop BGP neighbors, session parameter values are selected according to the following hierarchy:

- BFD session values configured at neighbor level
- BFD session values configured at BGP neighbor group level
- BFD session values configured at global BGP level
- Default values

For single-hop neighbors, interface-level BFD parameters are applied.

BFD sessions are deleted for a particular source or destination IPv4 or IPv6 address when a BGP session moves from the Established state to another BGP state.

BFD sessions are deleted when a BGP session for a neighbor is unconfigured locally. If you unconfigure a BGP session that is in an Established state and running a BFD session with a neighbor that is up, BGP sends a Cease message to the peer and deletes the BFD session. Upon receipt of this Cease message, the neighbor deletes the BFD session and moves to an Idle state.

Configuring BFD session parameters for BGP

BFD session parameters can be set globally for BGP-enabled interfaces.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

   ```
   device(config)# router bgp
   ```

4. Enter the **bfd interval** command with the **min-rx** and **multiplier** keywords to configure BFD session parameters globally for BGP-enabled interfaces.

   ```
   device(config-bgp-router)# bfd interval 110 min-rx 120 multiplier 15
   ```

   NOTE
   The **bfd interval** command is used for single-hop sessions only. Multihop sessions in BGP use either the values configured at the interface level using the **bfd interval** command or the default interval values.

   NOTE
   The **bfd interval** command is used for multihop sessions only. For single-hop sessions, configuration at the interface level is considered.

5. Enter the **bfd holdover-interval** command and specify a value to set the BFD holdover interval globally for BGP-enabled interfaces.

   ```
   device(config-bgp-router)# bfd holdover-interval 15
   ```

The following example configures BFD session parameters globally for BGP-enabled interfaces.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# bfd interval 110 min-rx 120 multiplier 15
device(config-bgp-router)# bfd holdover-interval 15
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# bfd interval 110 min-rx 120 multiplier 15
device(config-bgp-router)# bfd holdover-interval 15
```

Configuring BFD session parameters for BGP

BFD session parameters can be set globally for BGP-enabled interfaces.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **bfd-enable** command to enable BFD globally.

```
device(config-bgp)# bfd-enable
```

4. Enter the **bfd** command with the **min-tx**, **min-rx**, and **multiplier** keywords to configure BFD session parameters globally for BGP-enabled interfaces.

```
device(config-bgp)# bfd min-tx 110 min-rx 120 multiplier 15
```

NOTE
The **bfd** command is used for singlehop sessions only. Multihop sessions in BGP use either the values configured at interface level using the **bfd interval** command or the default interval values.

5. Enter the **bfd holdover-interval** command and specify a value to set the BFD holdover interval globally for BGP-enabled interfaces.

```
device(config-bgp)# bfd holdover-interval 15
```

The following example enables BFD globally and configures BFD session parameters for BGP-enabled interfaces.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# bfd-enable
device(config-bgp)# bfd min-tx 110 min-rx 120 multiplier 15
device(config-bgp)# bfd holdover-interval 15
```

Enabling BFD sessions for a specified BGP neighbor

BFD sessions can be configured for specified BGP neighbors.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **neighbor bfd** command, specifying an IP address, to enable BFD sessions for the specified neighbor.

```
device(config-bgp-router)# neighbor 10.10.1.1 bfd
```

5. Enter the **neighbor bfd** command, specifying an IP address, with the **interval**, **min-rx**, and **multiplier** keywords to set the BFD session timer values for the specified BGP neighbor.

```
device(config-bgp-router)# neighbor 10.10.1.1 bfd interval 120 min-rx 140 multiplier 10
```

NOTE
The **neighbor bfd interval** command is used for multihop sessions only. For single-hop sessions, configuration at the interface level is considered.

6. Enter the **neighbor bfd holdover-interval** command, specifying an IP address, and enter a value to set the BFD holdover interval for the specified BGP neighbor.

```
device(config-bgp-router)# neighbor 10.10.1.1 bfd holdover-interval 12
```

The following example enables a BFD session for a BGP neighbor with the IP address 10.10.1.1 and configures the BFD session parameters.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# neighbor 10.10.1.1 bfd
device(config-bgp-router)# neighbor 10.10.1.1 bfd interval 120 min-rx 140 multiplier 10
device(config-bgp-router)# neighbor 10.10.1.1 bfd holdover-interval 12

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# neighbor 10.10.1.1 bfd
device(config-bgp-router)# neighbor 10.10.1.1 bfd interval 120 min-rx 140 multiplier 10
device(config-bgp-router)# neighbor 10.10.1.1 bfd holdover-interval 12
```

Enabling BFD sessions for a specified BGP neighbor

BFD sessions can be configured for specified BGP neighbors.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **bfd-enable** command to enable BFD globally.

```
device(config-bgp)# bfd-enable
```

4. Enter the **neighbor bfd** command, specifying an IP address, with the **min-tx**, **min-rx**, and **multiplier** keywords to set the BFD session timer values for the specified BGP neighbor.

```
device(config-bgp)# neighbor 10.10.1.1 bfd min-tx 120 min-rx 140 multiplier 10
```

5. Enter the **neighbor bfd holdover-interval** command, specifying an IP address, and enter a value to set the BFD holdover interval for the specified BGP neighbor.

```
device(config-bgp)# neighbor 10.10.1.1 bfd holdover-interval 12
```

The following example configures the BFD session parameters for a BGP neighbor with the IP address 10.10.1.1.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# bfd-enable
device(config-bgp)# neighbor 10.10.1.1 bfd min-tx 120 min-rx 140 multiplier 10
device(config-bgp)# neighbor 10.10.1.1 bfd holdover-interval 12
```

Enabling BFD sessions for a specified BGP neighbor in a nondefault VRF

BFD sessions can be configured for specified BGP neighbors in a nondefault VRF instance.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   device(config)# router bgp
   ```

4. Enter the **address-family ipv4 unicast** command with the **vrf** keyword, specifying a VRF name, to enter BGP address-family IPv4 unicast VRF configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast vrf green
   ```

5. Enter the **neighbor bfd** command, specifying an IP address, to enable BFD sessions for the specified neighbor in a nondefault VRF instance.

   ```
   device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd
   ```

6. Enter the **neighbor bfd** command, specifying an IP address, with the **interval**, **min-rx**, and **multiplier** keywords to set the BFD session timer values for the specified BGP neighbor in a nondefault VRF instance.

   ```
   device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd interval 120 min-rx 140 multiplier 10
   ```

7. Enter the **neighbor bfd holdover-interval** command, specifying an IP address, and enter a value to set the BFD holdover interval for the specified BGP neighbor in a nondefault VRF instance.

   ```
   device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd holdover-interval 12
   ```

The following example enables a BFD session for a BGP neighbor with the IP address 10.10.1.1 and configures the BFD session parameters in VRF instance “green”.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast vrf green
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd interval 120 min-rx 140 multiplier 10
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd holdover-interval 12

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast vrf green
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd interval 120 min-rx 140 multiplier 10
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd holdover-interval 12
```

Enabling BFD sessions for a specified BGP neighbor in a nondefault VRF

BFD sessions can be configured for specified BGP neighbors in a nondefault VRF instance.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **address-family ipv4 unicast** command with the **vrf** keyword, specifying a VRF name, to enter BGP address-family IPv4 unicast VRF configuration mode.

   ```
   device(config-bgp)# address-family ipv4 unicast vrf green
   ```

4. Enter the **bfd-enable** command to enable BFD globally.

   ```
   device(config-bgp-ipv4u-vrf)# bfd-enable
   ```

5. Enter the **neighbor bfd** command, specifying an IP address, with the **min-tx**, **min-rx**, and **multiplier** keywords to set the BFD session timer values for the specified BGP neighbor.

   ```
   device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd min-tx 120 min-rx 140 multiplier 10
   ```

6. Enter the **neighbor bfd holdover-interval** command, specifying an IP address, and enter a value to set the BFD holdover interval for the specified BGP neighbor.

   ```
   device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd holdover-interval 12
   ```

7. Enter the **neighbor fail-over** command, specifying an IP address and using the **bfd-enable** keyword, to enable BFD support for failover for the BGP neighbor.

   ```
   device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 fail-over bfd-enable
   ```

The following example configures the BFD session parameters for a BGP neighbor with the IP address 10.10.1.1 and enables BFD support for failover for the BGP neighbor.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast vrf green
device(config-bgp-ipv4u-vrf)# bfd-enable
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd min-tx 120 min-rx 140 multiplier 10
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd holdover-interval 12
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 fail-over bfd-enable
```

Enabling BFD sessions for a specified BGP peer group

BFD sessions can be configured for specified BGP peer groups.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **neighbor** *peer-group-name* **peer-group** command to create a peer group.

```
device(config-bgp-router)# neighbor pg1 peer-group
```

5. Enter the **neighbor bfd** command, specifying a peer group, to enable BFD sessions for the specified peer group.

```
device(config-bgp-router)# neighbor pg1 bfd
```

6. Enter the **neighbor bfd** command, specifying a peer group, with the **interval**, **min-rx**, and **multiplier** keywords to set the BFD session timer values for the specified BGP peer group.

```
device(config-bgp-router)# neighbor pg1 bfd interval 200 min-rx 220 multiplier 25
```

7. Enter the **neighbor bfd holdover-interval** command, specifying a peer group, and enter a value to set the BFD holdover interval for the specified BGP peer group.

```
device(config-bgp-router)# neighbor pg1 bfd holdover-interval 17
```

The following example enables a BFD session for a BGP peer group called “pg1” and configures the BFD session parameters.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# neighbor pg1 peer-group
device(config-bgp-router)# neighbor pg1 bfd
device(config-bgp-router)# neighbor pg1 bfd interval 200 min-rx 220 multiplier 25
device(config-bgp-router)# neighbor pg1 bfd holdover-interval 17

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# neighbor pg1 peer-group
device(config-bgp-router)# neighbor pg1 bfd
device(config-bgp-router)# neighbor pg1 bfd interval 200 min-rx 220 multiplier 25
device(config-bgp-router)# neighbor pg1 bfd holdover-interval 17
```

Enabling BFD sessions for a specified BGP peer group

BFD sessions can be configured for specified BGP peer groups.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **bfd-enable** command to enable BFD globally.

```
device(config-bgp)# bfd-enable
```

4. Enter the **neighbor** *peer-group-name* **peer-group** command to create a peer group.

```
device(config-bgp)# neighbor pg1 peer-group
```

5. Enter the **neighbor bfd** command, specifying a peer group, with the **min-tx**, **min-rx**, and **multiplier** keywords to set the BFD session timer values for the specified BGP peer group.

```
device(config-bgp)# neighbor pg1 bfd min-tx 120 min-rx 140 multiplier 10
```

6. Enter the **neighbor bfd holdover-interval** command, specifying a peer group, and enter a value to set the BFD holdover interval for the specified BGP peer group.

```
device(config-bgp)# neighbor pg1 bfd holdover-interval 12
```

The following example configures the BFD session parameters for a BGP peer group called “pg1”.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# bfd-enable
device(config-bgp)# neighbor pg1 peer-group
device(config-bgp)# neighbor pg1 bfd min-tx 120 min-rx 140 multiplier 10
device(config-bgp)# neighbor pg1 bfd holdover-interval 12
```

Enabling BFD sessions for a specified BGP peer group in a nondefault VRF

BFD sessions can be configured for specified BGP peer groups in a nondefault VRF instance.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv6 unicast** command with the **vrf** keyword, specifying a VRF name, to enter BGP address-family IPv6 unicast VRF configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast vrf red
```

5. Enter the **neighbor** *peer-group-name* **peer-group** command to create a peer group.

```
device(config-bgp-ipv6u-vrf)# neighbor pg1 peer-group
```

6. Enter the **neighbor bfd** command, specifying a peer group, to enable BFD sessions for the specified peer group in a nondefault VRF instance.

```
device(config-bgp-ipv6u-vrf)# neighbor pg1 bfd
```

7. Enter the **neighbor bfd** command, specifying a peer group name, with the **interval**, **min-rx**, and **multiplier** keywords to set the BFD session timer values for the specified BGP neighbor in a nondefault VRF instance.

```
device(config-bgp-ipv6u-vrf)# neighbor pg1 bfd interval 145 min-rx 155 multiplier 15
```

8. Enter the **neighbor bfd holdover-interval** command, specifying a peer group name, and enter a value to set the BFD holdover interval for the specified BGP peer group in a nondefault VRF instance.

```
device(config-bgp-ipv6u-vrf)# neighbor pg1 bfd holdover-interval 18
```

The following example enables a BFD session for a BGP peer group called “pg1”and configures the BFD session parameters for VRF instance “red”.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast vrf red
device(config-bgp-ipv6u-vrf)# neighbor pg1 peer-group
device(config-bgp-ipv6u-vrf)# neighbor pg1 bfd
device(config-bgp-ipv6u-vrf)# neighbor pg1 bfd interval 145 min-rx 155 multiplier 15
device(config-bgp-ipv6u-vrf)# neighbor pg1 bfd holdover-interval 18
```

Enabling BFD sessions for a specified BGP peer group in a nondefault VRF

BFD sessions can be configured for specified BGP peer groups in a nondefault VRF instance.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv4 unicast** command with the **vrf** keyword, specifying a VRF name, to enter BGP address-family IPv4 unicast VRF configuration mode.

```
device(config-bgp)# address-family ipv4 unicast vrf green
```

4. Enter the **bfd-enable** command to enable BFD globally.

```
device(config-bgp-ipv4u-vrf)# bfd-enable
```

5. Enter the **neighbor** *peer-group-name* **peer-group** command to create a peer group.

```
device(config-bgp-ipv4u-vrf)# neighbor pg2 peer-group
```

6. Enter the **neighbor bfd** command, specifying a peer group, with the **min-tx**, **min-rx**, and **multiplier** keywords to set the BFD session timer values for the specified BGP peer group.

```
device(config-bgp-ipv4u-vrf)# neighbor pg2 bfd min-tx 130 min-rx 130 multiplier 12
```

7. Enter the **neighbor bfd holdover-interval** command, specifying a peer group, and enter a value to set the BFD holdover interval for the specified BGP peer group.

```
device(config-bgp-ipv4u-vrf)# neighbor pg2 bfd holdover-interval 14
```

8. Enter the **neighbor fail-over** command, specifying a peer group and using the **bfd-enable** keyword, to enable BFD support for failover for the BGP peer group.

```
device(config-bgp-ipv4u-vrf)# neighbor pg2 fail-over bfd-enable
```

The following example configures the BFD session parameters for a BGP peer group called “pg1” and enables BFD support for failover for the BGP neighbor.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast vrf green
device(config-bgp-ipv4u-vrf)# bfd-enable
device(config-bgp-ipv4u-vrf)# neighbor pg2 peer-group
device(config-bgp-ipv4u-vrf)# neighbor pg2 bfd min-tx 130 min-rx 130 multiplier 12
device(config-bgp-ipv4u-vrf)# neighbor pg2 bfd holdover-interval 14
device(config-bgp-ipv4u-vrf)# neighbor pg2 fail-over bfd-enable
```

BFD for OSPF

BFD support for OSPFv2 and OSPFv3 can be configured so that OSPF is a registered protocol with BFD and receives forwarding path detection failure messages from BFD.

BFD sessions can rapidly detect link faults and notify OSPF so that it quickly responds to network topology changes. BFD is supported for OSPF and is disabled by default.

Consider the following when configuring BFD for OSPF:

- OSPF uses single-hop BFD sessions.
- Virtual links are not supported.
- BFD for OSPF can be enabled in interface subtype configuration mode, OSPF VRF configuration mode, or OSPFv3 configuration mode. BFD must be enabled at both the interface level and global level to enable BFD for OSPF sessions.
- OSPF sends BFD a registration message even if BFD is not enabled on an interface or globally at the router OSPF level.
- Registration is global across all VRFs.
- BFD sessions are maintained across OSPF graceful restart.
- BFD for OSPF does not support authentication for BFD.

BFD for OSPF session creation and deletion

OSPF neighbors are discovered dynamically using the OSPF hello protocol. However, in the case of non-broadcast multiple access (NBMA) and point to multipoint (P2MP), neighbors must be manually configured. Regardless of whether neighbors are discovered dynamically or statically configured, all neighbors are associated with an interface. When BFD is enabled on an interface and at the global level, BFD sessions are created for all OSPF neighbors that are in greater than the 2-way state. A BFD session is created once it progresses to the INIT state.

Each interface can have its own BFD timers. OSPF does not influence the BFD timer value for the session. The configured BFD value, or the default value, is used by the BFD module when creating the session. A single-hop session is created for OSPF neighbors.

> **NOTE**
> When BFD for an OSPF session is configured, the normal OSPF hello mechanism is not disabled.

> **NOTE**
> OSPF neighbor sessions do not flap when BFD is enabled or disabled on the interface where the OSPF session is associated.

NOTE
OSPF assumes full connectivity between all systems on multi-access media such as LANs. If BFD is running on only a subset of systems on such a network, the assumptions of the control protocol may be violated, with unpredictable results.

OSPF BFD session deletion can happen in the following instances:

- If an OSPF neighbor session moves to a state below 2-way, OSPF triggers a BFD session deletion setting the reason as Path Down. When BFD receives this notification, it deletes the corresponding session. The remote BFD neighbor detects this as a detection timer expiry and propagates this session down to OSPF to terminate the session.
- If OSPF is disabled on an interface, all BFD sessions associated with each neighbor are deleted.
- If BFD is administratively disabled on an interface, BFD reports this event as a port change notification to OSPF. Upon receipt of this notification, a BFD session deletion for each neighbor is sent and BFD removes these sessions if no other client is using the same sessions. BFD communicates this change to the remote peer. When the remote BFD peer receives this Remote Admin Down event, it propagates this event to the OSPF protocol. OSPF does not provide any action for this event.

Enabling BFD on a specified OSPFv2-enabled interface

BFD sessions can be configured on one or more OSPFv2-enabled interfaces.

BFD sessions are initiated on specified OSPFv2-enabled interfaces only if BFD is enabled globally using the **bfd all-interfaces** command in OSPF VRF configuration mode.

BFD sessions are initiated on specified OSPFv2-enabled interfaces only if BFD is enabled globally using the **bfd** command in OSPF VRF configuration mode.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface** command and specify an interface.

   ```
   device(config)# interface ethernet 1/1
   device(config)# interface tengigabitethernet 101/0/10
   device(config)# interface ethernet 0/4
   device(config)# interface ethernet 1/4
   ```

3. Enter the **ip ospf bfd** command to enable BFD on the specified interface.

   ```
   device(config-if-e1000-1/1)# ip ospf bfd
   device(config-if-te-101/0/10)# ip ospf bfd
   device(config-if-eth-0/4)# ip ospf bfd
   device(config-if-eth-1/4)# ip ospf bfd
   ```

The following example enables BFD on a specified OSPFv2-enabled Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/1
device(config-if-e1000-1/1)# ip ospf bfd
```

The following example enables BFD on a specified OSPFv2-enabled 10-gigabit Ethernet interface.

```
device# configure terminal
device(config)# tengigabitethernet 101/0/10
device(config-if-te-101/0/10)# ip ospf bfd
```

The following example enables BFD on a specified OSPFv2-enabled Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 0/4
device(config-if-eth-0/4)# ip ospf bfd

device# configure terminal
device(config)# interface ethernet 1/4
device(config-if-eth-1/4)# ip ospf bfd
```

Configuring BFD for OSPFv2 globally

BFD for OSPFv2 can be configured globally on interfaces where BFD has been configured.

BFD must be configured using the **ip ospf bfd** command on each OSPFv2 interface to initiate BFD sessions.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config)# router ospf

   device(config-rbridge-id-122)# router ospf
   ```

4. Enter the **bfd all-interfaces** command to enable BFD globally.

   ```
   device(config-ospf-router)# bfd all-interfaces
   ```

5. Enter the **bfd** command to enable BFD globally.

   ```
   device(config-router-ospf-vrf-default-vrf)# bfd
   ```

6. Enter the **bfd holdover-interval** command to set the BFD holdover interval.

   ```
   device(config-ospf-router)# bfd holdover-interval 12

   device(config-router-ospf-vrf-default-vrf)# bfd holdover-interval 12
   ```

The following example enables BFD globally and sets the BFD holdover interval to 12.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# bfd all-interfaces
device(config-ospf-router)# bfd holdover-interval 12

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# bfd
device(config-router-ospf-vrf-default-vrf)# bfd holdover-interval 12

device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# bfd
device(config-router-ospf-vrf-default-vrf)# bfd holdover-interval 12
```

Configuring BFD for OSPFv2 globally in a nondefault VRF instance

BFD for OSPFv2 can be configured globally on interfaces where BFD has been configured in nondefault VRF instances.

BFD must be configured using the **ip ospf bfd** command on each OSPFv2 interface to initiate BFD sessions.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command and specify a VRF name to enable OSPFv2 in a nondefault VRF instance.

   ```
   device(config-rbridge-id-122)# router ospf vrf red
   device(config)# router ospf vrf red
   ```

4. Enter the **bfd** command to enable BFD.

   ```
   device(config-router-ospf-vrf-red)# bfd
   ```

5. Enter the **bfd holdover-interval** command to set the BFD holdover interval.

   ```
   device(config-router-ospf-vrf-red)# bfd holdover-interval 12
   ```

The following example enables BFD globally and sets the BFD holdover interval to 12 for VRF “red”.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf vrf red
device(config-router-ospf-vrf-red)# bfd
device(config-router-ospf-vrf-red)# bfd holdover-interval 12

device# configure terminal
device(config)# router ospf vrf red
device(config-router-ospf-vrf-red)# bfd
device(config-router-ospf-vrf-red)# bfd holdover-interval 12
```

Enabling BFD on a specified OSPFv3-enabled interface

BFD sessions can be configured on one or more OSPFv3-enabled interfaces.

BFD sessions are initiated on specified OSPFv3-enabled interfaces only if BFD is enabled globally using the **bfd** command in OSPFv3 VRF configuration mode.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface** command and specify an interface.

   ```
   device(config)# interface ve 24
   ```

3. Enter the **ipv6 ospf bfd** command to enable BFD on the specified OSPFv3-enabled interface.

   ```
   device(config-ve-24)# ipv6 ospf bfd
   device(config-if-Ve-24)# ipv6 ospf bfd
   ```

The following example enables BFD on an OSPFv3-enabled virtual Ethernet (VE) interface.

```
device# configure terminal
device(config)# interface ve 24
device(config-ve-24)# ipv6 ospf bfd

device# configure terminal
device(config)# interface ve 24
device(config-if-Ve-24)# ipv6 ospf bfd
```

Enabling BFD on a specified OSPFv3-enabled interface

BFD sessions can be configured on one or more OSPFv3-enabled interfaces.

BFD sessions are initiated on specified OSPFv3-enabled interfaces only if BFD is enabled globally using the **bfd** command in OSPFv3 VRF configuration mode.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface** command and specify an interface.

   ```
   device(config)# interface ethernet 1/1
   ```

3. Enter the **ipv6 ospf bfd** command to enable BFD on the specified OSPFv3-enabled interface.

   ```
   device(config-if-e1000-1/1)# ipv6 ospf bfd
   ```

The following example enables BFD on an OSPFv3-enabled Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/1
device(config-if-e1000-1/1)# ipv6 ospf bfd
```

Configuring BFD for OSPFv3 globally

BFD for OSPFv3 can be configured globally on interfaces where BFD has been configured.

BFD must be configured using the **ipv6 ospf bfd** command on each OSPFv3 interface to initiate BFD sessions.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3.

   ```
   device(config-rbridge-id-122)# ipv6 router ospf

   device(config)# ipv6 router ospf
   ```

4. Enter the **bfd** command to enable BFD globally.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# bfd
   ```

5. Enter the **bfd all-interfaces** command to enable BFD for all interfaces.

```
device(config-ospf6-router)# bfd all-interfaces
```

6. Enter the **bfd holdover-interval** command to set the BFD holdover interval.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# bfd holdover-interval 20

device(config-ospf6-router)# bfd holdover-interval 20
```

The following example enables BFD and sets the BFD holdover interval to 20.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# bfd
device(config-ipv6-router-ospf-vrf-default-vrf)# bfd holdover-interval 20

device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# bfd all-interfaces
device(config-ospf6-router)# bfd holdover-interval 20

device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# bfd
device(config-ipv6-router-ospf-vrf-default-vrf)# bfd holdover-interval 20
```

Configuring BFD for OSPFv3 globally in a nondefault VRF instance

BFD for OSPFv3 can be configured globally on interfaces where BFD has been configured in nondefault VRF instances.

BFD must be configured using the **ipv6 ospf bfd** command on each OSPFv3 interface to initiate BFD sessions.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 router ospf** command and specify a VRF name to enable OSPFv3 in a nondefault VRF instance.

```
device(config-rbridge-id-122)# ipv6 router ospf vrf orange

device(config)# ipv6 router ospf vrf orange
```

4. Enter the **bfd** command to enable BFD.

```
device(config-ipv6-router-ospf-vrf-orange)# bfd
```

5. Enter the **bfd holdover-interval** command to set the BFD holdover interval.

```
device(config-ipv6-router-ospf-vrf-orange)# bfd holdover-interval 22
```

The following example enables BFD and sets the BFD holdover interval to 22 for VRF “orange”.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf vrf orange
device(config-ipv6-router-ospf-vrf-orange)# bfd
device(config-ipv6-router-ospf-vrf-orange)# bfd holdover-interval 22
```

```
device# configure terminal
device(config)# ipv6 router ospf vrf orange
device(config-ipv6-router-ospf-vrf-orange)# bfd
device(config-ipv6-router-ospf-vrf-orange)# bfd holdover-interval 22
```

BFD for IS-IS

BFD support for IS-IS (for both IPv4 and IPv6 IS-IS neighbors) can be configured so that IS-IS is a registered protocol with BFD. BFD support for the IS-IS protocol can be configured globally or for specific interfaces.

> NOTE
> You cannot configure BFD for an IS-IS session when one side of the IS-IS adjacency is using IPv4 only and the other side is using IPv6 only.

Enabling BFD on a specified IS-IS-enabled interface

BFD sessions can be configured on one or more IS-IS-enabled interfaces.

BFD sessions are initiated on specified IS-IS interfaces only if BFD is enabled globally using the **bfd** command in ISIS router configuration mode.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface** command and specify an interface.

   ```
   device(config)# interface ethernet 1/4
   ```

3. Enter the **isis bfd** command to enable BFD on the specified interface.

   ```
   device(config-if-e1000-1/4)# isis bfd
   ```

   ```
   device(config-if-eth-1/4)# isis bfd
   ```

The following example enables BFD on a specified IS-IS-enabled Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/4
device(config-if-e1000-1/4)# isis bfd
```

```
device# configure terminal
device(config)# interface ethernet 1/4
device(config-if-eth-1/4)# isis bfd
```

Configuring BFD for IS-IS globally

BFD for IS-IS can be configured globally on interfaces where BFD has been configured.

BFD must be configured using the **isis bfd** command on each IS-IS interface to initiate BFD sessions.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter ISIS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **bfd** command with the **all-interfaces** keyword to enable BFD globally.

   ```
   device(config-isis-router)# bfd all-interfaces
   ```

4. Enter the **bfd** command to enable BFD globally.

   ```
   device(config-isis-router)# bfd
   ```

5. Enter the **bfd holdover-interval** command, and enter a value, to set the BFD holdover interval.

   ```
   device(config-isis-router)# bfd holdover-interval 12
   ```

The following example enables BFD globally and sets the BFD holdover interval to 12.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# bfd all-interfaces
device(config-isis-router)# bfd holdover-interval 12

device# configure terminal
device(config)# router isis
device(config-isis-router)# bfd
device(config-isis-router)# bfd holdover-interval 12
```

BFD for VXLAN extension tunnels

BFD support for Virtual Extensible LAN (VXLAN) extension tunnels can be configured for diagnostic purposes.

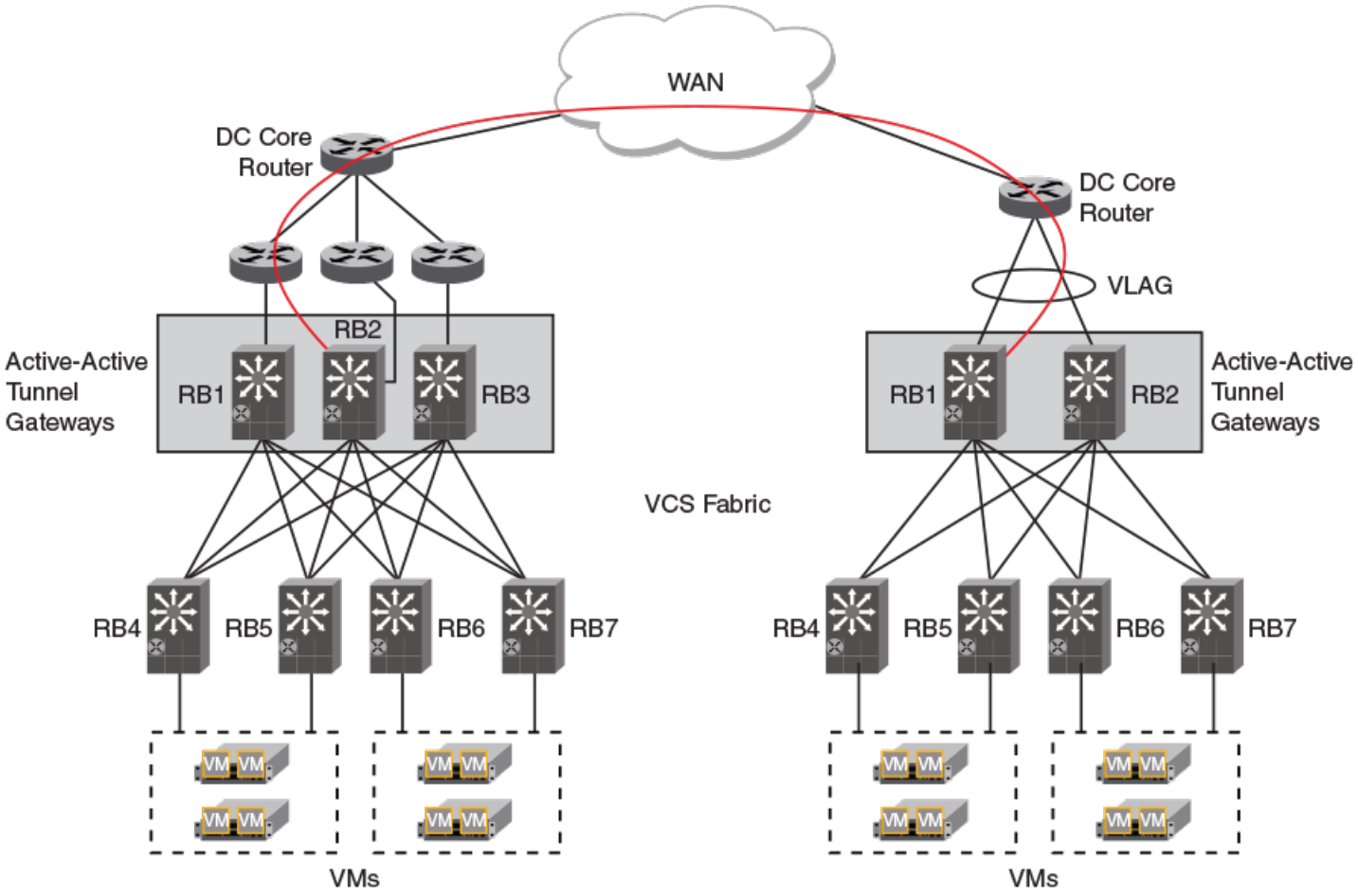
VXLAN extension tunnels facilitate a Layer 3 overlay to extend Layer 2 VLANs across VCS clusters. Two different VCS clusters can be connected by a VXLAN tunnel. A VXLAN tunnel can have its end points in both VCS clusters, with the possibility of multiple Layer 3 ECMP paths (four paths) between the endpoints. BFD has the ability to monitor the tunnel for reachability and quick fault detection.

> **NOTE**
> RBridges in a VXLAN Network Identifier (VNI) tunnel endpoint (VTEP) are assumed to be connected symmetrically to the destination VTEP. BFD sessions run from all RBridge in the VTEP.

A single BFD session is used to monitor the connectivity of a VXLAN tunnel endpoint. When a tunnel session is created, an RBridge is selected as the “master RBridge”. This master RBridge runs the full BFD state machine for the VXLAN tunnel. BFD session information is also created in other RBridges, but these backup RBridges do not run the full state machine for that specific BFD session. The RBridge with the lowest RBridge ID is selected as the Master RBridge. Furthermore, a newly joined RBridge with a lower RBridge ID than the current master RBridge can pre-empt the mastership from the existing master.

In the figure below, RB1 is the master RBridge for the tunnel session established between the source VTEP and the destination VTEP. RBridges RB2 and RB3 are designated as backup RBridges. The nodes are connected through Inter-Switch Link (ISLs). The participants of the VTEP gateway (RB1, RB2, and RB3) communicate with each other using fabric messages that are transmitted through ISL links.

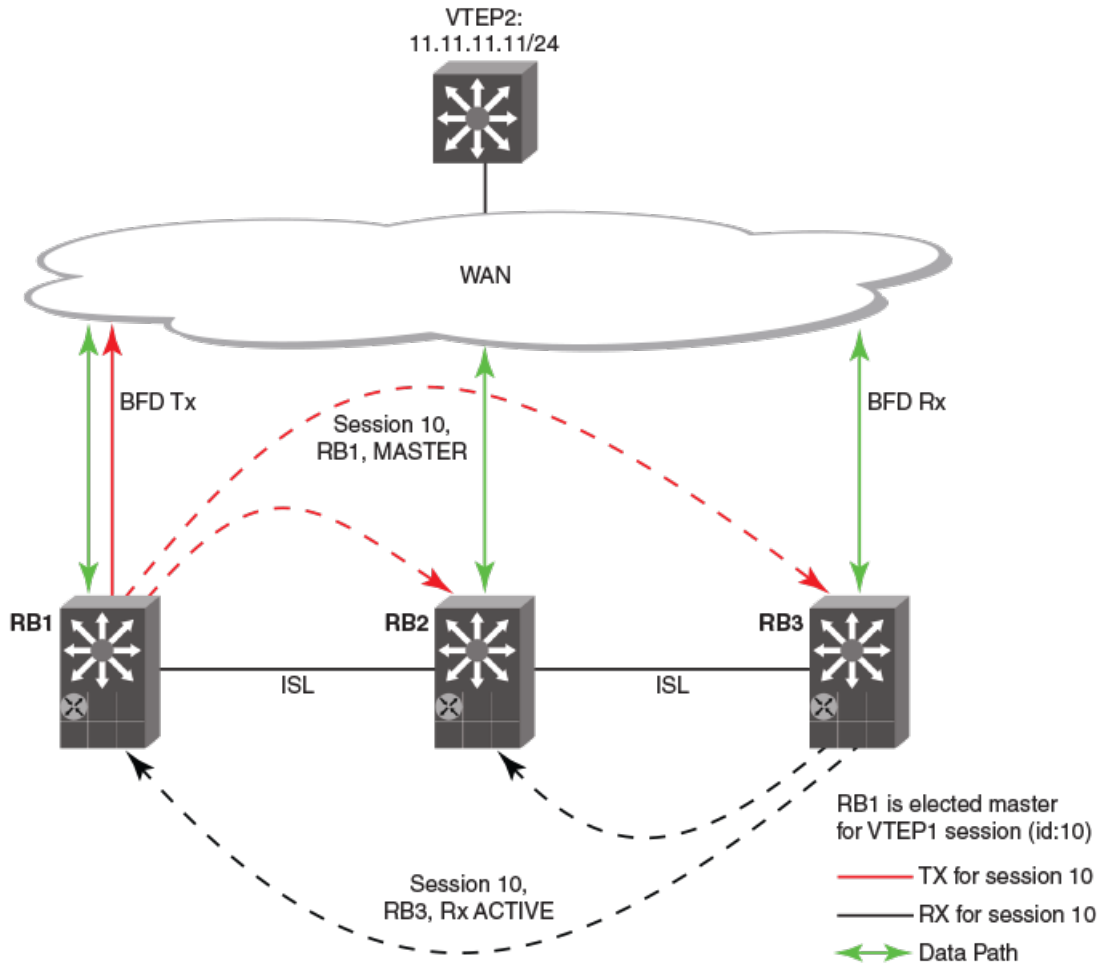
FIGURE 25 BFD for VXLAN extension tunnels



Once a session is established, RB1 continues to transmit BFD control packets based on the negotiated interval. BFD control packets can arrive on any of the gateway RBridges, including the initiator. If control packets become trapped on an RBridge, BFD updates the local forwarding path state to Active, and propagates this information to all RBridges in the cluster.

The figure below shows a BFD session in a cluster where there is at least one communication path between the gateways and the destination host. The session database for all RBridges contains at least one active entry.

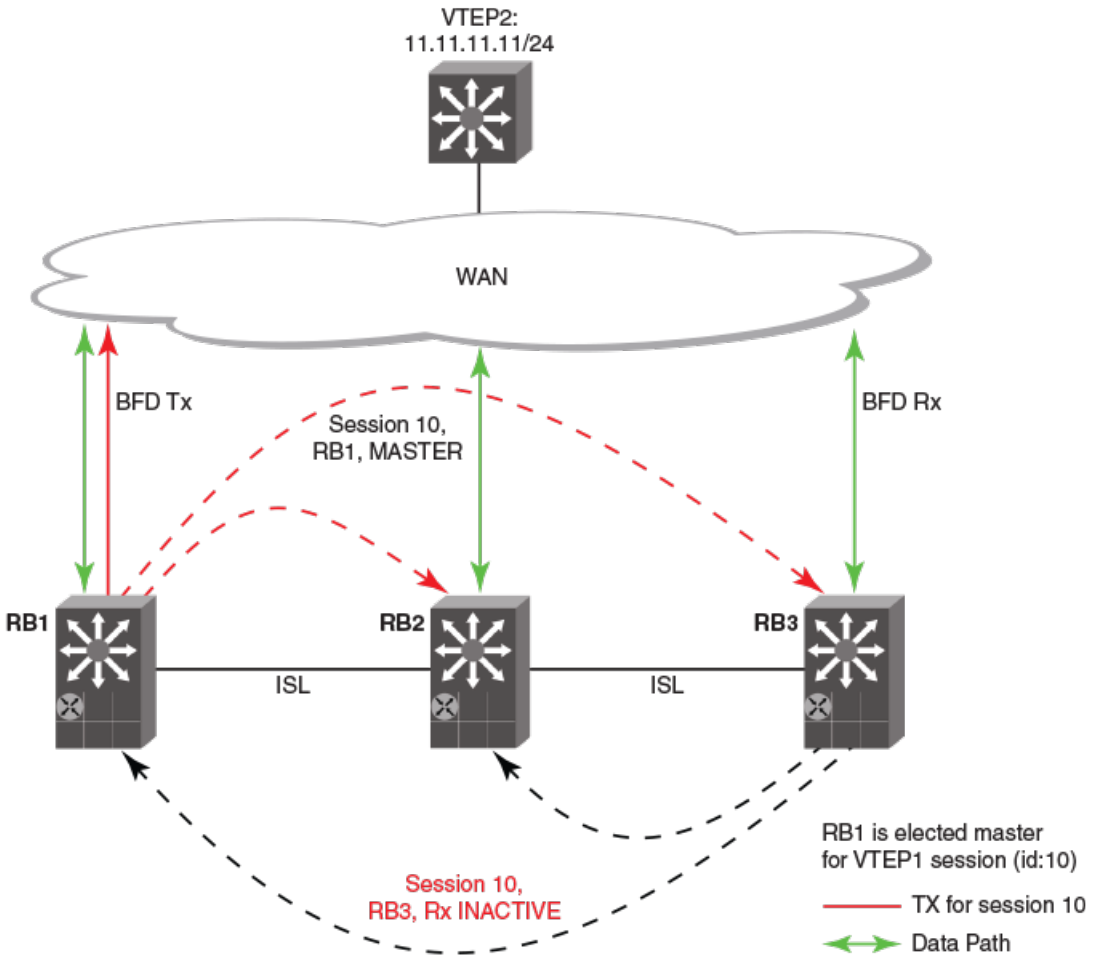
FIGURE 26 Control flow of a BFD Session in an Up state



Session ID	SRC IP	DST IP	Master (Tx)	Rx			Final State
				RB1	RB2	RB3	
10	10.10.10.10/24	10.11.11.11/24	RB1	0	0	1	1

The figure below shows a BFD session in a cluster where all communication paths between the gateways and destination host are lost. The session database for all RBridges contains no active entries.

FIGURE 27 BFD session in a down state



Session ID	SRC IP	DST IP	Master (Tx)	Rx			Final State
				RB1	RB2	RB3	
10	10.10.10.10/24	10.11.11.11/24	RB1	0	0	0	0

A session is declared as down only if the state is declared down on all RBridges. As long as one forwarding path to the destination exists from any of the RBridges in a cluster, a BFD session remains active in the entire fabric.

A maximum of 20 BFD sessions can be configured for extension tunnels.

Configuring BFD on a VXLAN extension tunnel

BFD can be configured on a VXLAN extension tunnel.

BFD support for Virtual Extensible LAN (VXLAN) extension tunnels can be configured for diagnostic purposes. VXLAN extension tunnels facilitate a Layer 3 overlay to extend Layer 2 VLANs across VCS clusters. Two different VCS clusters can be connected by a VXLAN tunnel. A VXLAN tunnel can have its end points in both VCS clusters, with the possibility of multiple Layer 3 ECMP paths (four paths) between the endpoints. BFD has the ability to monitor the tunnel for reachability and quick fault detection. VXLAN extension tunnels facilitate a Layer 3 overlay to extend Layer 2 VLANs across VCS clusters. Two different VCS clusters can be connected by a VXLAN tunnel. A VXLAN tunnel can have its end points in both VCS clusters, with the possibility of multiple Layer 3 ECMP paths (four paths) between the endpoints. BFD has the ability to monitor the tunnel for reachability and quick fault detection.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **overlay-gateway** command and specify a name to create a VXLAN overlay gateway instance and enter VXLAN overlay gateway configuration mode.

   ```
   device(config)# overlay-gateway gateway1
   ```

3. Enter the **type** command using the **layer2-extension** keyword to specify that the VXLAN overlay gateway uses a Layer 2 extension.

   ```
   device(config-overlay-gw-gateway1)# type layer2-extension
   ```

4. Enter the **ip interface** command using the **loopback** keyword to specify an IPv4 loopback interface.

   ```
   device(config-overlay-gw-gateway1)# ip interface loopback 22
   ```

5. Enter the **site** command and specify a name to create a remote Layer 2 extension site in a VXLAN overlay gateway context and enable VXLAN overlay gateway site configuration mode.

   ```
   device(config-overlay-gw-gateway1)# site s1
   ```

6. Enter the **ip address** command to specify the destination IPv4 address of the tunnel.

   ```
   device(config-site-s1)# ip address 10.11.12.13
   ```

7. Enter the **bfd** command to enable BFD on the VXLAN overlay gateway site.

   ```
   device(config-site-s1)# bfd
   ```

8. Enter the **bfd interval** command with the **min-rx** and **multiplier** keywords to set the BFD session parameters on the VXLAN overlay gateway site.

   ```
   device(config-site-s1)# bfd interval 2000 min-rx 3000 multiplier 26
   ```

The following example enables BFD on the VXLAN overlay gateway site and sets the BFD session parameters.

```
device# configure terminal
device(config)# overlay-gateway gateway1
device(config-overlay-gw-gateway1)# type layer2-extension
device(config-overlay-gw-gateway1)# ip interface loopback 22
device(config-overlay-gw-gateway1)# site s1
device(config-site-s1)# ip address 10.11.12.13
device(config-site-s1)# bfd
device(config-site-s1)# bfd interval 2000 min-rx 3000 multiplier 26
```

BFD for NSX tunnels

BFD support for NSX tunnels can be configured on the VDX 6740.

An NSX Controller expects a VXLAN Network Identifier (VNI) tunnel endpoint (VTEP) to use BFD to determine the reachability of NSX replicator VTEPs at the end of the tunnels. The VTEP is expected to use this information to determine whether the tunnel is a healthy NSX replicator for sending egress BUM traffic.

In an NSX logical network there can be multiple NSX replicators for redundancy. When there are multiple replicator tunnels, the switch can distribute the BUM traffic across all such tunnels by assigning a different set of VLANs to each of the tunnels. Egress BUM traffic on different VLANs are now forwarded through different tunnels, increasing the overall efficiency of the network. The switch can receive ingress BUM traffic over any VLAN on any NSX replicator tunnel. When a tunnel is deleted or a BFD " down" state is detected, the system automatically reassigns its BUM VLANs to other available replicator tunnels. Refer to the *VXLAN NSX replicator load balancing* section of the*Extreme Network OS Layer 2 Switching Configuration Guide* for more information.

Details of the BUM-enabled NSX replicator tunnel can be printed using the **show tunnel** command. Refer to the *Extreme Network OS Command Reference* for more information on this command.

> NOTE
> BFD parameters are not configurable in switches for VXLAN
> NSX.

For more information on NSX tunnels, refer to the *Extreme Network OS Layer 2 Switching Configuration Guide*.

> NOTE
> Open vSwitch hardware_vtep v1.3.0 schema support is required on VTEP switches for using BFD in a NSX logical network.

BFD for static routes

Unlike dynamic routing protocols, such as OSPF and BGP, static routing has no method of peer discovery and fault detection. BFD for IPv4 and IPv6 static routes provides rapid detection of failure in the bidirectional forwarding path between BFD peers.

BFD for static routes allows you to detect failures that impact the forwarding path of a static route. This feature supports both single-hop and multihop BFD static routes for both IPv4 and IPv6. Unless the BFD session is up, the gateway for the static route is considered unreachable, and the affected routes are not installed in the routing table. BFD can remove the associated static route from the routing table if the next-hop becomes unreachable indicating that the BFD session has gone down.

Static routes and BFD neighbors are configured separately. A static route is automatically associated with a static BFD neighbor if the static route's next-hop exactly matches the neighbor address of the static BFD neighbor and BFD monitoring is enabled for the static route.

When a static BFD neighbor is configured, BFD asks the routing table manager (RTM) if there is a route to the neighbor. If a route exists, and if the route is directly connected, then BFD initiates a single-hop session. If the route is not directly connected, BFD establishes a multi-hop session. Once the session comes up, BFD adds the corresponding static routes to RTM. If no route exists, then BFD will not add the corresponding static routes to RTM.

When a static BFD neighbor is configured, BFD asks the routing table manager if there is a route to the BFD neighbor. If a route exists, and if the next-hop is directly connected, BFD initiates a single-hop session. If the next-hop is not directly connected, BFD establishes a multihop session.

When a BFD session goes down because the BFD neighbor is no longer reachable, static routes monitored by BFD are removed from the routing table manager. The removed routes can be added back if the BFD neighbor becomes reachable again. Single-hop BFD

sessions use the BFD timeout values configured on the outgoing interface. Timeout values for multihop BFD sessions are specified along with each BFD neighbor. Multiple static routes going to the same BFD neighbor use the same BFD session and timeout values.

BFD considerations and limitations for static routes

There are a number of things to consider when configuring BFD for IPv4 and IPv6 static routes.

- BFD is not supported on interface-based static routes because BFD requires that the next-hop address matches the address of the BFD neighbor.
- Only one static route BFD session for a neighbor is created at any instance. This is always based on the best path for the neighbor.
- Static BFD for a multihop BFD neighbor reachable by way of an Equal-Cost Multi-Path (ECMP) is not supported. Static BFD must be configured explicitly for each next-hop corresponding to each path.
- When an interface goes down, multihop IPv4 static route sessions are not deleted. Multihop IPv6 static route sessions are deleted.
- BFD for static routes is supported in both Local-only mode and Distributed mode.
- BFD sessions can be single-hop or multihop sessions.
- A BFD multihop session is supported for a next-hop resolved through OSPF or BGP.
- If a BFD session goes down and the BFD neighbor had Layer 3 direct connectivity, associated static routes are removed from the routing table so that data packets can use the available alternate path.
- If a BFD neighbor is not directly connected and a BFD session goes down, associated static routes are removed only if an alternate path to the neighbor exists.
- BFD for static routes is supported in both default and nondefault VRFs.
- BFD for IPv6 static routes is supported in both associated and unassociated mode.
- BFD for Link-local IPv6 addresses is supported.
- When configuring BFD for Link-local IPv6 static routes, the source IPv6 address must be link-local and an interface must be provided.

BFD for static routes configuration

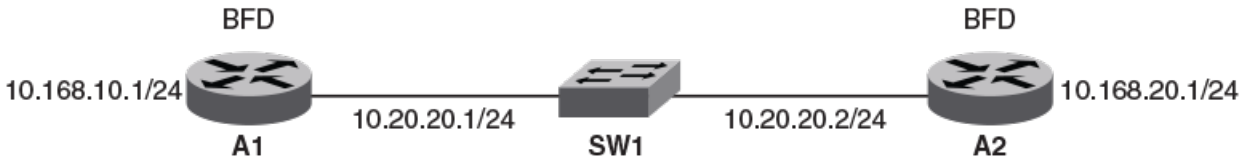
Single-hop BFD IPv4 static route sessions use the timer values configured for the outgoing interface to the directly connected neighbors. Multihop BFD IPv4 static route sessions use the timer values configured using the **ip route static bfd** and **ip route static bfd holdover-interval** commands. If the timer values configured conflict with the timer values set for BGP for the same next-hop, BFD uses the smaller value to meet the more stringent requirement.

Single-hop BFD IPv6 static route sessions use the timer values configured for the outgoing interface to the directly connected neighbors. Multihop BFD IPv6 static route sessions use the timer values configured using the **ipv6 route static bfd** and **ipv6 route static bfd holdover-interval** commands. If the timer values configured conflict with the timer values set for BGP for the same next-hop, BFD uses the smaller value to meet the more stringent requirement.

If you remove a static BFD session, the corresponding session is removed by BFD without removing static routes from the routing table and ongoing traffic is not disrupted. If a BFD session goes down because a BFD neighbor is no longer reachable, all associated static routes are removed from the routing table. Existing traffic on these static routes is interrupted.

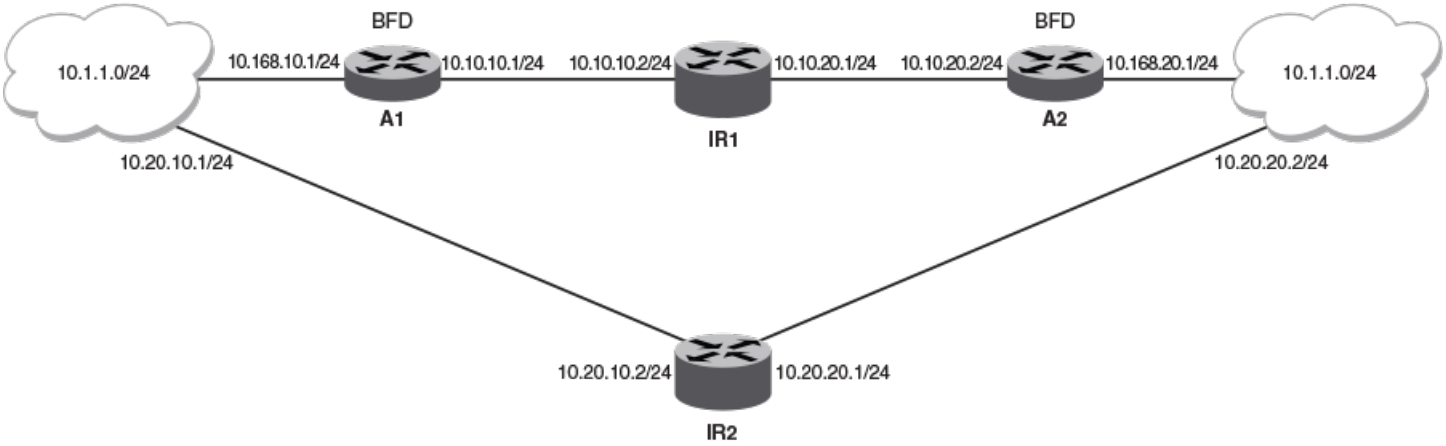
The following figure shows a single-hop static BFD session. A1 has a static route to 10.168.20.0/24 with the next-hop as 10.20.20.2. A2 has a static route to 10.168.10.0/24 with the next-hop as 10.20.20.1. A switch is connected between the routers. BFD can be configured to monitor next-hop 10.20.20.2 on A1 and 10.20.20.1 on A2.

FIGURE 28 Single-hop static BFD session



The following figure shows a multihop static BFD session. Static routes are configured on A1 to reach the 10.1.1.0/24 subnet by way of 10.168.20.1. 10.168.20.1 is reachable by way of the intermediate routers IR1 and IR2. A static route is configured on A2 to reach the 10.1.1.0/24 subnet by way of 10.168.10.1. This next-hop is reachable in turn by way of the intermediate routers IR1 and IR2. If one BFD session goes down, the corresponding route is removed from the routing table and the data packets take another path.

FIGURE 29 Multihop ECMP static BFD session



NOTE
When configuring BFD for static routes, static routes are already installed in the routing table and traffic is running on those static routes. When you configure BFD on these static routes, a similar BFD configuration also occurs on BFD neighbors. If BFD session creation fails or a BFD session does not come up, associated static routes are not removed from the routing table; therefore ongoing traffic on these static routes is not interrupted. A BFD session may not be established if a neighbor is busy or if the maximum number of sessions has been reached on the neighbor. Ongoing traffic on installed static routes is not interrupted.

Configuring BFD on an IP static route

BFD can be configured globally on IP static routes. Repeat the steps in this procedure on each BFD neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip route static bfd** command, specifying a source and destination IP address, and use the **interval**, **min-rx**, and **multiplier** keywords to configure BFD session parameters on an IP static route.

```
device(config-rbridge-id-122)# ip route static bfd 10.0.2.1 10.1.1.1 interval 500 min-rx 500
multiplier 5
```

```
device(config)# ip route static bfd 10.0.2.1 10.1.1.1 interval 500 min-rx 500 multiplier 5
```

4. Enter the **ip route-static bfd** command, specifying a source and destination IP address, and use the **interval**, **min-rx**, and **multiplier** keywords to configure BFD session parameters on an IP static route.

```
device(config)# ip route static-bfd 10.0.2.1 10.1.1.1 interval 500 min-rx 500 multiplier 5
```

5. Enter the **ip route static bfd holdover-interval** command and specify an interval to set the BFD holdover interval globally for IP static routes.

```
device(config-rbridge-id-122)# ip route static bfd holdover-interval 15
```

```
device(config)# ip route static bfd holdover-interval 15
```

The following example configures BFD session parameters on an IP static route where the destination IP address is 10.0.2.1 and the source IP address is 10.1.1.1. The BFD holdover interval is set globally to 15 for IP static routes.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)#)# ip route static bfd 10.0.2.1 10.1.1.1 interval 500 min-rx 500 multiplier 5
device(config-rbridge-id-122)# ip route static bfd holdover-interval 15
```

```
device# configure terminal
device(config)# ip route static bfd 10.0.2.1 10.1.1.1 interval 500 min-rx 500 multiplier 5
device(config)# ip route static bfd holdover-interval 15
```

The following example configures BFD session parameters on an IP static route where the destination IP address is 10.0.2.1 and the source IP address is 10.1.1.1.

```
device# configure terminal
device(config)# ip route static-bfd 10.0.2.1 10.1.1.1 interval 500 min-rx 500 multiplier 5
```

Configuring BFD on an IP static route in a nondefault VRF instance

BFD can be configured on IP static routes in a nondefault VRF instance. Repeat the steps in this procedure on each BFD neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **vrf** command and specify a VRF name to enter Virtual Routing and Forwarding (VRF) configuration mode.

```
device(config-rbridge-id-122)# vrf green
```

```
device(config)# vrf green
```

4. Enter the **address-family ipv4 unicast** command to enter IPv4 address-family unicast configuration mode..

```
device(config-vrf-green)# address-family ipv4 unicast
```

5. Enter the **address-family ipv4** command to enter IPv4 address-family configuration mode..

```
device(config-vrf-green)# address-family ipv4 unicast
```

6. Enter the **ip route static bfd** command, specifying a source and destination IP address, and use the **interval**, **min-rx**, and **multiplier** keywords to configure BFD session parameters on an IP static route in a nondefault VRF instance.

```
device(vrf-ipv4-unicast)# ip route static bfd 10.0.0.1 10.1.1.2 interval 500 min-rx 500 multiplier 5
```

7. Enter the **ip route static-bfd** command, specifying a source and destination IP address, and use the **interval**, **min-rx**, and **multiplier** keywords to configure BFD session parameters on an IP static route in a nondefault VRF instance.

```
device(config-vrf-green-ipv4)# ip route static-bfd 10.0.0.1 10.1.1.2 interval 500 min-rx 500
multiplier 5
```

The following example configures BFD session parameters on an IP static route in a nondefault VRF instance, where the destination IP address is 10.0.0.1 and the source IP address is 10.1.1.2.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# vrf green
device(config-vrf-green)# address-family ipv4 unicast
device(vrf-ipv4-unicast)# ip route static bfd 10.0.0.1 10.1.1.2 interval 500 min-rx 500 multiplier 5

device# configure terminal
device(config)# vrf green
device(config-vrf-green)# address-family ipv4 unicast
device(vrf-ipv4-unicast)# ip route static bfd 10.0.0.1 10.1.1.2 interval 500 min-rx 500 multiplier 5

device# configure terminal
device(config)# vrf green
device(config-vrf-green)# address-family ipv4
device(config-vrf-green-ipv4)# ip route static-bfd 10.0.0.1 10.1.1.2 interval 500 min-rx 500 multiplier 5
```

Configuring BFD on an IPv6 static route

BFD can be configured globally on IPv6 static routes. Repeat the steps in this procedure on each BFD neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 route static bfd** command, specifying a source and destination IPv6 address and an interface type, and use the **interval**, **min-rx**, and **multiplier** keywords to configure BFD session parameters on an IPv6 static route.

```
device(config-rbridge-id-122)# ipv6 route static bfd fe80::a fe80::b ve 20 interval 100 min-rx 100
multiplier 10

device(config)# ipv6 route static bfd fe80::a fe80::b ve 20 interval 100 min-rx 100 multiplier 10
```

4. Enter the **ipv6 route static-bfd** command, specifying a source and destination IPv6 address and an interface type, and use the **interval**, **min-rx**, and **multiplier** keywords to configure BFD session parameters on an IPv6 static route.

```
device(config)# ipv6 route static-bfd fe80::a fe80::b 20 interval 100 min-rx 100 multiplier 10
```

5. Enter the **ipv6 route static bfd holdover-interval** command and specify an interval to set the BFD holdover interval globally for IPv6 static routes.

```
device(config-rbridge-id-122)# ipv6 route static bfd holdover-interval 25

device(config)# ipv6 route static bfd holdover-interval 25
```

The following example configures BFD session parameters on an IPv6 static route where the destination IPv6 address is fe80::a and the source IPv6 address is fe80::b. A VE interface is specified and the BFD holdover interval is set globally to 25 for IPv6 static routes.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)#)# ipv6 route static bfd fe80::a fe80::b ve 20 interval 100 min-rx 100
multiplier 10
device(config-rbridge-id-122)# ipv6 route static bfd holdover-interval 25

device# configure terminal
device(config)# ipv6 route static bfd fe80::a fe80::b ve 20 interval 100 min-rx 100 multiplier 10
device(config)# ipv6 route static bfd holdover-interval 25

device# configure terminal
device(config)# ipv6 route static-bfd fe80::a fe80::b 20 interval 100 min-rx 100 multiplier 10
```

Configuring BFD on an IPv6 static route in a nondefault VRF instance

BFD can be configured on IPv6 static routes in a nondefault VRF instance. Repeat the steps in this procedure on each BFD neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **vrf** command and specify a VRF name to enter Virtual Routing and Forwarding (VRF) configuration mode.

```
device(config-rbridge-id-122)# vrf blue

device(config)# vrf blue
```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address-family unicast configuration mode..

```
device(config-vrf-blue)# address-family ipv6 unicast
```

5. Enter the **address-family ipv6** command to enter IPv6 address-family configuration mode..

```
device(config-vrf-blue)# address-family ipv6 unicast
```

6. Enter the **ipv6 route static bfd** command, specifying a source and destination IPv6 address and an interface type, and use the **interval**, **min-rx**, and **multiplier** keywords to configure BFD session parameters on an IPv6 static route in a nondefault VRF instance.

```
device(vrf-ipv6-unicast)# ipv6 route static bfd fe70::a fe60::b ve 10 interval 1000 min-rx 2000
multiplier 20
```

7. Enter the **ipv6 route static-bfd** command, specifying a source and destination IPv6 address and an interface type, and use the **interval**, **min-rx**, and **multiplier** keywords to configure BFD session parameters on an IPv6 static route in a nondefault VRF instance.

```
device(config-vrf-blue-ipv6)# ipv6 route static-bfd fe70::a fe60::b interval 1000 min-rx 2000
multiplier 20
```

The following example configures BFD session parameters on an IPv6 static route in a nondefault VRF instance, where the destination IPv6 address is fe80::a and the source IPv6 address is fe80::b.

```
device# configure terminal
device(config)# vrf blue
device(config-vrf-blue)# address-family ipv6
device(config-vrf-blue-ipv6)# ipv6 route static bfd fe70::a fe60::b interval 1000 min-rx 2000 multiplier 20
```

The following example configures BFD session parameters on an IPv6 static route in a nondefault VRF instance, where the destination IPv6 address is fe80::a and the source IPv6 address is fe80::b. A VE interface is specified.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# vrf blue
device(config-vrf-blue)# address-family ipv6 unicast
device(vrf-ipv6-unicast)# ipv6 route static bfd fe70::a fe60::b ve 10 interval 1000 min-rx 2000 multiplier
20

device# configure terminal
device(config)# vrf blue
device(config-vrf-blue)# address-family ipv6 unicast
device(vrf-ipv6-unicast)# ipv6 route static bfd fe70::a fe60::b ve 10 interval 1000 min-rx 2000 multiplier
20
```

BFD for RSVP-TE LSP

BFD provides a mechanism to detect data plane failure for MPLS LSP in the order of sub-second. Unlike MPLS LSP ping where MPLS control plane can be verified against the data plane, BFD is used only to detect data plane failure. There are a couple advantages in using BFD instead of LSP ping for MPLS data plane failure detection. BFD provides a faster failure detection mechanism because it does not require control plane verification. BFD can also be used to detect faults for a large number of LSPs without manual trigger for each LSP the way LSP ping does.

BFD session for RSVP LSP is very similar to the BFD session set up for ISIS and OSPF with the following differences:

- MPLS BFD session is bootstrapped using LSP ping.
- The IP TTL for transmitted MPLS BFD control packets from ingress LSR to egress LSR must be set to 1 instead of 255.
- After MPLS BFD session is up, the local discriminator and the source IP address are not allowed to change without bringing down the MPLS BFD session.
- The transmit and receive portion of the session can be on different LPs because the LSP is unidirectional and the returned path from egress to ingress LSR depends on IP routing.

MPLS BFD is set up between the ingress and egress LSR. The MPLS BFD session uses an asynchronous operating mode without echo function. BFD demands an operating mode and echo function that are not currently defined for MPLS BFD. Authentication is not supported (it is currently not supported for OSPF or ISIS either).

BFD configuration on the MLX Series and XMR Series devices are provided at the global MPLS configuration level and at the LSP level as follows:

- The global configuration provides you a convenient way to enable and disable BFD for all LSPs that have BFD enabled (this is similar to how it is provided for ISIS and OSPF). In addition, the you can change the default settings for transmit and receive

intervals and detection time multiplier to be used for all BFD sessions. On egress LSR, this global setting is used to decide whether a BFD session starts upon receiving MPLS echo request with BFD Discriminator TLV and the time intervals to be included in the BFD control packet are to be sent back to the ingress LSR.

- Under the LSP configuration, you are able to enable or disable BFD and change the default settings for minimum transmit and receive intervals as well as detection time multiple. If those parameters are not specified, the values from the global configuration are used.

BFD can be enabled or disabled without program exit at the global MPLS level or for each individual LSP without affecting the LSP operational status. In addition, the BFD parameters can also be changed without program exit This does not change the state of the BFD session.

One BFD session is created for each non-redundant LSP. BFD session is associated with the active path which can be normal or protected, or detour path. For redundant LSP, a separate BFD session is created for the currently active secondary path.

BFD session creation for RSVP-TE LSP

On ingress, one BFD session is created for each LSP. A BFD session is created after the LSP comes UP. The BFD session status is displayed as part of the show LSP output. When the BFD session is not UP, a failure reason is displayed as part of the show LSP detail output. Possible failure reasons include exceeding the maximum number of BFD sessions the system can support or the global BFD configuration is disabled. In the case where BFD session is not brought up because BFD packet from the egress LSR is not received, the MPLS Echo Request with the BFD Discriminator TLV is resent until the session is UP. The retry timer is exponentially backed off.

On egress, a BFD session is created after an MPLS Echo Request is received with the BFD Discriminator TLV and the MPLS BFD enabled globally and the maximum number of BFD sessions has not been reached yet. The BFD session created on egress LSR also is counted towards the maximum number of BFD sessions. When the number of BFD sessions has reached a maximum, neither the MPLS Echo Reply nor the BFD control packet are sent. The ingress LSR will retry.

Because the source IP address cannot be changed for MPLS BFD session after the session has comes up, the LSR-ID is used as the source IP address for all MPLS BFD packets. This guarantees that the session does not go down when the LSP path switch occurs.

FRR LSP

Only one BFD session is created for a FRR LSP. When a switchover from protected to detour path occurs, and when the detour is on a different LP, the BFD session is moved to the LP where the detour path resides. The BFD session can go down when the LP already has a maximum number of BFD sessions running. When the detour is on the same LP, the outgoing interface and label stack is updated on the existing LP.

A BFD session is not created for detour path originating on a transit LSR.

Redundant LSP

One BFD session is created for a primary path of a redundant LSP. When the secondary path is hot-standby, a separate BFD session is created for it if and only if BFD is enabled for a secondary path. The two sessions operate independently.

Adaptive LSP

When a new instance of an adaptive LSP comes UP, a BFD session automatically moves to a new LP when the new instance is created on a different LP. Otherwise, the local outgoing interface and label stack updates on the existing LP. When the BFD session needs to be moved to a different LP, it is possible that the BFD session may be down when the LP already has maximum number of BFD sessions running.

BFD session deletion for RSVP-TE LSP

A BFD session is deleted when any of the following events happen:

- LSP goes down
- BFD is disabled for the LSP
- BFD is disabled for all LSP (example: through global MPLS configuration)

BFD session modification for RSVP-TE LSP

You can change the BFD parameters globally or for an individual LSP without program shutdown without affecting the operational status of the LSP and the BFD session. When you change the global configuration, the change is applied to all egress MPLS BFD sessions and only to those ingress BFD sessions whose parameters are derived from the global configuration.

BFD session down handling for RSVP-TE LSP

When a BFD session for a LSP goes down on ingress LSR because BFD detection time has expired, it may trigger path switchover if possible (protected to detour or primary to secondary path switchover). In the case where there is no alternative path, the LSP is brought down and the BFD session is deleted. The LSP goes through the normal retry mechanism in order to come back UP.

On egress LSR, BFD session down does not have any impact on the RSVP session.

BFD for RSVP-TE LSPs configuration

BFD can be configured for use with RSVP-LSPs to detect data plane failures for MPLS LSPs. Although the LSP ping facility can also be used for this purpose, BFD provides the following advantages:

- BFD provides faster failure detection because it does not required control plane verification, which is required by LSP ping.
- BFD can be used to detect faults on a large number of LSPs without requiring manual interaction, which is required by LSP ping.

BFD configuration for RSVP-TE LSPs is performed at the global and LSP levels, as described:

- **BFD for RSVP-TE LSPs global configuration** - allows you to enable and disable BFD on all of the RSVP-TE LSPs that have been configured for BFD at the LSP level. In addition, use the global command to set revised default values for the transmit interval, receive interval, and for the detection time multiplier for all BFD sessions on RSVP-TEs. This configuration command can also be used as a convenient method to turn BFD for MPLS on or off.
- **BFD for RSVP-TE LSPs configuration at the LSP level** - allows you to enable and disable BFD for individual RSVP-TE LSPs. You can also change the values for the transmit interval, receive interval, and for the detection time multiplier from the default values. If these values are not specified at this level, they are obtained from the values configured at the global level.

BFD, which is disabled by default, can be enabled or disabled at the global MPLS level, or for each individual LSP without affecting the LSP operational status. BFD parameters can also be changed without changing the state of the BFD session.

BFD for RSVP-TE LSPs operates with Fast ReRoute (FRR), Redundant, and Adaptive LSPs as described:

- **FRR LSPs -** Only one BFD session is created for an FRR LSP. A separate BFD session is not created for the detour path. When a switchover from a protected to a detour path occurs, the detour path resides on another interface module, and the BFD session is moved to that interface module. The BFD session can go down if the interface module has already reached the maximum number of BFD sessions. If the detour path is on the same interface module, the outgoing interface and label stack are updated on that interface module. A BFD session is not created for a detour path originated on a transit LSR.

- **Redundant LSPs -** One BFD session is created for the primary path of a redundant LSP. If the secondary path is in the hot-standby condition, a separate BFD session is created for it, but only if BFD is enabled on the secondary path. The two sessions operate independently.
- **Adaptive LSPs -** If a new instance of an adaptive LSP comes up on a different interface module, its BFD session is automatically created on that module. Otherwise, the local outgoing interface and label stack are updated on the existing interface module. When a BFD session is moved to a different interface module, the BFD session may be brought down if the interface module has already reached the maximum number of BFD sessions allowed on it.

BFD session support per-router and per-interface module

There is a limit to the number of BFD sessions available on a per-router and per-interface module basis as described:

- **per-router** - A maximum number of 250 BFD sessions are permitted per device
- **per-interface module** - A maximum number of 80 BFD sessions (Tx or Rx) are permitted per-interface module

These limitations are inclusive of any BFD sessions created for OSPFv2 or v3 and IS-IS. If creating a BFD session will exceed these limits, the session will be denied.

BFD session down behavior

When a BFD session for an LSP goes down on an ingress LSR because the BFD detection time has expired, one of the following path switchovers will be triggered; from the protected path to the detour path, or from the primary path to the secondary path. In configurations with no alternative path, the LSP is brought down and the BFD session is deleted. The LSP then follows the normal retry procedures to come back up. On an egress LSR, a down BFD session does not have any impact on the RSVP session.

AdminDown State

The AdminDown mechanism in BFD is intended to signal that the BFD session is being taken down for administrative purposes, and the session state is not indicative of the activity of the data path. Therefore, a system should not indicate a connectivity failure to a client if either the local session state or the remote session state (if known) transitions to AdminDown when that client has an independent means of activity detection (typically, control protocols).

If a client does not have any independent means of activity detection, a system should indicate a connectivity failure to a client, and assume the semantics of Down state, if either the local or remote session state transitions to AdminDown. Otherwise, the client will not be able to determine whether the path is viable, if not unfortunate results may occur.

Reaction to BFD Session State Changes

If a BFD session transitions from Up state to AdminDown, or the session transitions from Up to Down because the remote system is indicating that the session is in state AdminDown, clients should not take any control protocol action.

Enabling BFD for RSVP-TE LSPs at the global level

When using BFD for RSVP-TE LSPs, you must configure BFD globally at the **router mpls** level.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **router mpls** command to enter MPLS configuration mode.

```
device(config)# router mpls
```

3. Enter the **bfd** command to enable BFD globally.

```
device(config-router-mpls)# bfd
```

4. Enter the **min-tx** command with the **min-rx** and **multiplier** keywords and enter a value to set new values for the transmit interval, receive interval, and for the detection time multiplier.

```
device(config-mpls-bfd)# min-tx 500 min-rx 500 multiplier 5
```

The following example enables BFD globally for RSVP-TE LSPs and sets new values for the transmit interval, receive interval, and for the detection time multiplier.

```
device# configure terminal
device(config)# router mpls
device(config-router-mpls)# bfd
device(config-mpls-bfd)# min-tx 500 min-rx 500 multiplier 5
```

Enabling BFD for a specific RSVP-TE-LSP

When you configure BFD globally, you must also configure it locally for the individual LSPs on which you want it to operate. You can also set separate values for the transmit interval, receive interval, and for the detection time multiplier for the specified LSP. The following example enables BFD for the LSP named “blue” and sets new parameter values.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **router mpls** command to enter MPLS configuration mode.

```
device(config)# router mpls
```

3. Enter the **lsp** command and specify a name to access LSP subconfiguration mode.

```
device(config-mpls)# lsp blue
```

4. Enter the **bfd** command to enable BFD.

```
device(config-mpls-lsp-blue)# bfd
```

5. Enter the **min-tx** command with the **min-rx** and **multiplier** keywords and enter a value to set new values for the transmit interval, receive interval, and for the detection time multiplier.

```
device(config-mpls-lsp-blue-bfd)# min-tx 500 min-rx 500 multiplier 5
```

The following example enables BFD for the LSP named “blue” and sets new parameter values.

```
device# configure terminal
device(config)# router mpls
device(config-mpls)# lsp blue
device(config-mpls-lsp-blue)# bfd
device(config-mpls-lsp-blue-bfd)# min-tx 500 min-rx 500 multiplier 5
```

Configuring BFD for the secondary path of an LSP

You can configure BFD for the secondary path of an LSP.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **router mpls** command to enter MPLS configuration mode.

   ```
   device(config)# router mpls
   ```

3. Enter the **lsp** command and specify a name to access LSP subconfiguration mode.

   ```
   device(config-mpls)# lsp blue
   ```

4. Enter the **secondary-path** command and specify a name to set a secondary explicit path.

   ```
   device(config-mpls-lsp-blue)# secondary-path alt_sf_to_sj
   ```

5. Enter the **bfd** command to enable BFD.

   ```
   (config-mpls-lsp-blue-sec-path)# bfd
   ```

The following example configures BFD for the secondary path of an LSP.

```
device# configure terminal
device(config)# router mpls
device(config-mpls)# lsp blue
device(config-mpls-lsp-blue)# secondary-path alt_sf_to_sj
(config-mpls-lsp-blue-sec-path)# bfd
```

Enabling the IP router alert option

You can set the IP router alert option for MPLS BFD packets sent from the ingress device to the egress device. NetIron OS devices support the IP router alert option as defined in RFC 2113.

This feature is supported only for XMR Series and MLX Series devices.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **router mpls** command to enter MPLS configuration mode.

   ```
   device(config)# router mpls
   ```

3. Enter the **lsp** command and specify a name to access LSP subconfiguration mode.

   ```
   device(config-mpls)# lsp blue
   ```

4. Enter the **bfd** command to enable BFD.

   ```
   device(config-mpls-lsp-blue)# bfd
   ```

5. Enter the **set-router-alert-option** command to set the IP router alert option for MPLS BFD packets sent from the ingress device to the egress device.

   ```
   device (config-mpls-lsp-blue-bfd)#set-router-alert-option
   ```

The following example enables the router alert option under the LSP BFD.

```
device# configure terminal
device(config)# router mpls
device(config-mpls)# lsp blue
device(config-mpls-lsp-blue)# bfd
device (config-mpls-lsp-blue-bfd)# set-router-alert-option
```

Displaying BFD information

Various **show** commands verify information about BFD configurations.

Use one or more of the following commands to verify BFD information. The commands do not have to be entered in this order.

1. Enter the **show bfd** command.

```
device> show bfd

BFD State: ENABLED Version: 1 Use PBIF Assist: Y
Current Registered Protocols: ospf/0  ospf6/0
All Sessions: Current: 4 Maximum Allowed: 100 Maximum Exceeded Count: 0
LP Sessions: Maximum Allowed on LP: 40  Maximum Exceeded Count for LPs: 0
LP Tx/Rx Sessions LP Tx/Rx Sessions LP Tx/Rx Sessions LP Tx/Rx Sessions
1   4/4           2   2/2            3  0/0            4  0/0
5   0/0           6   0/0            7  0/0            8  0/0
9   0/0           10  0/0            11 0/0            12 0/0
13  0/0           14  0/0            15 0/0            16 0/0
BFD Enabled ports count: 2
Port       MinTx       MinRx       Mult Sessions
eth 2/1    100         100         3    2
eth 3/1    100         100         3    2
```

   This example output displays BFD information.

2. Enter the **show bfd neighbors** command.

```
device> show bfd neighbors

Total number of Neighbor entries: 2
NeighborAddress          State   Interface Holddown  Interval  R/H
10.14.1.1                UP      eth 3/1   300000    100000    Y/S
10.2.1.1                 UP      eth 2/1   300000    100000    Y/S
```

   This example output displays BFD neighbor information for the default VRF.

3. Enter the **show bfd neighbors applications** command.

```
device> show bfd applications

Registered Protocols Count: 3
Protocol  VRFID     Parameter HoldoverInterval
isis      0         0         2
ospf6     0         1         10
ospf      0         0         5
```

   This example displays BFD registered protocol information for the device.

4. Enter the **show bfd neighbors interface** command with the **ethernet** keyword, and specify an interface

```
device> show bfd neighbors interface ethernet 1/1

 BFD State: ENABLED Version: 1 Use PBIF Assist: Y SH setup delay 180 MH setup delay 0
 Current Registered Protocols: mpls/0  ospf/2  ospf6/0  ospf/4  ospf/0
 All Sessions: Current: 0 Maximum Allowed: 250 Maximum Exceeded Count: 0
 Maximum TX/RX Sessions Allowed on LP: 80  Maximum Session Exceeded Count for LPs: 0
   LP Tx/Rx Sessions LP Tx/Rx Sessions LP Tx/Rx Sessions LP Tx/Rx Sessions
   1   0/0           2   0/0           3   0/0           4   0/0
 BFD Enabled ports count: 1
 Port       MinTx       MinRx       Mult Sessions
 eth 1/1    55          55          5    0
```

This example shows detailed neighbor information about a specified Ethernet interface.

5. Enter the **show bfd neighbors isis** command.

```
device> show bfd neighbors isis

Total Entries:1 R:RxRemote(Y:Yes/N:No)H:Hop(S:Single/M:Multi)
NeighborAddress       State  Interface       Holddown   Interval  R/H
10.40.40.10           UP     eth 3/6         900000     300000    Y/S
```

This example shows BFD neighbor information for IS-IS.

6. Enter the **show bfd neighbors ospf** command.

```
device> show bfd neighbors ospf

Total Entries:1 R:RxRemote(Y:Yes/N:No)H:Hop(S:Single/M:Multi)
NeighborAddress                State  Interface       Holddown   Interval  R/H
1.1.1.1                        UP     eth 1/2         300000     100000    Y/S
```

This example shows BFD neighbor information for OSPFv2.

7. Enter the **show bfd neighbors ospf6** command.

```
device> show bfd neighbors ospf6

Total Entries:1 R:RxRemote(Y:Yes/N:No)H:Hop(S:Single/M:Multi)
NeighborAddress                        State  Interface       Holddown   Interval  R/H
fe80::21b:edff:fe3b:8601               UP     eth 1/2         300000     100000    Y/S
```

This example shows BFD neighbor information for OSPFv3.

8. Enter the **show bfd neighbors static** command.

```
device> show bfd neighbors static

Total Entries:1 R:RxRemote(Y:Yes/N:No)H:Hop(S:Single/M:Multi)
NeighborAddress                State  Interface       Holddown   Interval  R/H
1.1.1.1                        UP     eth 1/2         300000     100000    Y/S
```

This example shows BFD neighbor information for IP static routes.

Displaying BFD information

Various **show** commands verify information about BFD configurations.

Use one or more of the following commands to verify BFD information. The commands do not have to be entered in this order.

1. Enter the **show bfd** command.

```
device# show bfd

Rbridge-id:1
  BFD State: ENABLED, Version: 1
  Supported Protocols: static-ip, tunnel, ospf6, ospf
  All Sessions: Current: 6 Max Allowed: 250 Max Exceeded Count: 0

  Port       MinTx       MinRx       Mult Sessions
  ====       =====       =====       ==== ========
  Te 1/0/9   50          50          3    1
  Te 1/0/10  50          50          3    1
  Tu 61441   1000        1000        3    1
```

   This example output displays BFD information on a stackable switch.

2. Enter the **show bfd** command.

```
device# show bfd
RBridge: 1
BFD State: ENABLED, Version: 1
Supported Protocols: ospf, Tunnel
All Sessions: Current: 4 Max Allowed: 100 Max Exceeded Count: 0
Agent Sessions: Max Allowed on LC: 40 Max Exceeded Count for LCs: 0

LC   Tx/Rx Sessions   LC  Tx/Rx Sessions  LC  Tx/Rx Sessions  LC  Tx/Rx Sessions
1    4/4              2   2/2             3   0/0             4   0/0
5    0/0              6   0/0             7   0/0             8   0/0

BFD Enabled ports count: 2
Port       MinTx    MinRx    Mult   Sessions
Te 1/2/1   100      100      3       2
Tunnel 1   100      100      3       2
```

   This example output displays BFD information on a chassis.

3. Enter the **show bfd neighbors** command.

```
device# show bfd neighbors
OurAddr                    NeighAddr                   State    Int        Rbridge-id
=======                    =========                   =====    ===        ==========
10.10.10.1                 10.10.10.2                  UP       Ve 10      1
fe80::52eb:1aff:fe13:91d   fe80::52eb:1aff:fe13:c5ed   UP       Ve 11      1
12.12.12.1                 12.12.12.2                  UP       Ve 12      1
2001::1                    2001::2                     DOWN     Ve 13      1
```

   This example output displays BFD neighbor information for the default VRF. To obtain information about non-default VRF instances, a non-default VRF instance must be specified. Refer to the *Network OS Command Reference* for more information.

4. Enter the **show bfd neighbors application** command with the **ospf** and **details** keywords.

```
device# show bfd neighbors application ospf details

OurAddr            NeighAddr        State    Int          Rbridge-id
=======            =========        =====    ===          ==========
5.0.0.1            5.0.0.2          UP       Te 1/0/10    1

  Local    State: UP            Diag: 0              Demand mode: 0    Poll: 0

  Received State: UP            Diag: 0              Demand mode: 0    Poll: 0
  Final: 1
  Local    MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Received MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Rx Count: 265660              Tx Count: 275613
  LD/RD:          1/1           Heard from Remote: Y
  Current Registered Protocols: ospf
  Uptime: 0 day 3 hour 50 min 53 sec 400 msec

OurAddr            NeighAddr        State    Int          Rbridge-id
=======            =========        =====    ===          ==========
2.0.0.1            2.0.0.2          UP       Te 1/0/9     1

  Local    State: UP            Diag: 0              Demand mode: 0    Poll: 0

  Received State: UP            Diag: 0              Demand mode: 0    Poll: 0
  Final: 1
  Local    MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Received MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Rx Count: 265559              Tx Count: 275550
  LD/RD:          2/3           Heard from Remote: Y
  Current Registered Protocols: ospf
  Uptime: 0 day 3 hour 50 min 50 sec 248 msec
```

This example displays detailed information about neighbor OSPF sessions.

5. Enter the **show bfd neighbors dest-ip** command with the **details** keyword.

```
device# show bfd neighbors dest-ip 1.1.1.6 details
OurAddr            NeighAddr        State    Int          Rbridge-id
=======            =========        =====    ===          ==========
1.1.1.5            1.1.1.6          UP       Ve 5         5

  Local    State: UP            Diag: 0              Demand mode: 0    Poll: 0
  Received State: UP            Diag: 0              Demand mode: 0    Poll: 0    Final: 1
  Local    MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Received MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Rx Count: 226354              Tx Count: 234323
  LD/RD:          1/11          Heard from Remote: Y
  Current Registered Protocols: ospf
  Uptime: 0 day 3 hour 4 min 48 sec 248 msec
```

This example shows detailed neighbor information about a destination device.

6. Enter the **show bfd neighbors interface** command with the **ve** and **details** keywords.

```
device# show bfd neighbors interface ve 5 details
OurAddr            NeighAddr        State    Int          Rbridge-id
=======            =========        =====    ===          ==========
1.1.1.5            1.1.1.6          UP       Ve 5         5

  Local    State: UP            Diag: 0              Demand mode: 0    Poll: 0
  Received State: UP            Diag: 0              Demand mode: 0    Poll: 0    Final: 1
  Local    MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Received MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Rx Count: 225447              Tx Count: 233383
  LD/RD:          1/11          Heard from Remote: Y
  Current Registered Protocols: ospf
  Uptime: 0 day 3 hour 4 min 0 sec 208 msec
```

This example shows detailed neighbor information about a specified VE interface.

7. Enter the **show bfd neighbors vrf** command and specify a VRF name.

```
device# show bfd neighbors vrf default-vrf

OurAddr            NeighAddr        State    Int          Rbridge-id
=======            =========        =====    ===          ==========

1.1.1.5             1.1.1.6         UP       Ve 5         5
```

This example shows BFD neighbor information for the default VRF.

8. Enter the **show ip bgp neighbors** command.

```
device# show ip bgp neighbors

    Total number of BGP Neighbors: 1
1   IP Address: 100.1.1.2, AS: 100 (IBGP), RouterID: 19.19.19.19, VRF: default-vrf
    State: ESTABLISHED, Time: 0h6m49s, KeepAliveTime: 60, HoldTime: 180
       KeepAliveTimer Expire in 44 seconds, HoldTimer Expire in 136 seconds
    Minimal Route Advertisement Interval: 0 seconds
       RefreshCapability: Received
    Messages:    Open    Update  KeepAlive Notification Refresh-Req
       Sent    : 1       0       9         0            0
       Received: 1       0       8         0            0
    Last Update Time: NLRI       Withdraw          NLRI       Withdraw
                  Tx: ---        ---           Rx: ---        ---
    Last Connection Reset Reason:Unknown
    Notification Sent:     Unspecified
    Notification Received: Unspecified
    Neighbor NLRI Negotiation:
      Peer Negotiated IPV4  unicast  capability
      Peer configured for IPV4 unicast  Routes
    Neighbor ipv6 MPLS Label Capability Negotiation:
    Neighbor AS4 Capability Negotiation:
    Outbound Policy Group:
       ID: 4, Use Count: 1
    BFD:Enabled,BFDSessionState:UP,Multihop:No
        LastBGP-BFDEvent:RX:Up,BGP-BFDError:No Error
        HoldOverTime(sec) Configured:0,Current:0,DownCount:0
    TCP Connection state: ESTABLISHED, flags:00000033 (0,0)
     Maximum segment size: 1460
    TTL check: 0, value: 0, rcvd: 64
       Byte Sent:   216, Received: 197
       Local host:  100.1.1.1, Local  Port: 8177
       Remote host: 100.1.1.2, Remote Port: 179
```

This output shows BGP neighbor information including configured BFD information.

9. Enter the **show ip ospf** command.

```
device# show ip ospf

OSPF Version                  Version 2
Router Id                     19.19.19.19
ASBR Status                   No
ABR Status                    No         (0)
Redistribute Ext Routes from
Initial SPF schedule delay    0          (msecs)
Minimum hold time for SPFs    0          (msecs)
Maximum hold time for SPFs    0          (msecs)
External LSA Counter          0
External LSA Checksum Sum     00000000
Originate New LSA Counter     4
Rx New LSA Counter            5
External LSA Limit            14447047
Database Overflow Interval    0
Database Overflow State :     NOT OVERFLOWED
RFC 1583 Compatibility :      Enabled
Slow neighbor Flap-Action :   Disabled,   timer 300
Nonstop Routing:              Disabled
Graceful Restart:             Disabled,   timer 120
Graceful Restart Helper:      Enabled
BFD:                          Enabled
LDP-SYNC: Not globally enabled
Interfaces with LDP-SYNC enabled: None
```

This output displays OSPF information including configured BFD information.

10. Enter the **show ip ospf** command with the **extensive** keyword.

```
device# show ip ospf neighbor extensive
Number of Neighbors is 1, in FULL state 1
Port      Address        Pri State     Neigh Address   Neigh ID    Ev      Opt Cnt
Ve 10     10.10.10.1     1   FULL/DR   10.10.10.2      2.2.2.2     6        2   0
  Neighbor is known for 0d:00h:01m:34s and up for 0d:00h:00m:55s
  Neighbor BFD State:UP, BFD HoldoverInterval(sec):Configured:2 Current:0
```

This output displays detailed information about all neighbors including configured BFD information.

11. Enter the **show ipv6 ospf neighbor detail** command.

```
device# show ipv6 ospf neighbor detail
Total number of neighbors in all states: 1
Number of neighbors in state Full      : 1

RouterID        Pri State    DR              BDR             Interface     [State]
2.2.2.2           1 Full     2.2.2.2         1.1.1.1         Ve 11         [BDR]
                Option: 00-00-13    QCount: 0    Timer: 686
BFD State: UP, BFD HoldoverInterval(sec):Configured: 3 Current: 0
```

This output displays detailed OSPF neighbor information including configured BFD information.

12. Enter the **show tunnel replicator** command.

```
device# show tunnel replicator
Tunnel 61441, mode VXLAN, rbridge-ids 1-2
Ifindex 2080436225, Admin state up, Oper state up, BFD up
Overlay gateway "k", ID 1
Source IP 50.50.50.3 ( Ve 555, Vrid 1 ), Vrf default-vrf
Destination IP 20.20.20.251
Active next hops on rbridge 1:
    IP: 20.20.20.251, Vrf: default-vrf
    Egress L3 port: Ve 20, Outer SMAC: 0005.3365.381f
    Outer DMAC: 0050.5682.48e4
    Egress L2 Port: Po 200, Outer ctag: 20, stag:0, Egress mode: Local
    BUM forwarder: yes
Packet count: RX 20701        TX 264784
Byte count  : RX (NA)            TX 62217299
```

This example displays NSX service node information, including configured BFD status.

Displaying BFD information

Various **show** commands verify information about BFD configurations.

Use one or more of the following commands to verify BFD information. The commands do not have to be entered in this order.

1. Enter the **show bfd** command.

```
device# show bfd

BFD State: ENABLED, Version: 1
Current Registered Protocols: ospf, Tunnel
All Sessions: Current: 4 Max Allowed: 250 Max Exceeded Count: 0
Agent Sessions: Max Allowed on LC: 40 Max Exceeded Count for LCs: 0
Port      MinTx   MinRx   Mult  Sessions
Eth 0/1/2 100     100     3     2
```

This example output displays global BFD information.

2. Enter the **show bfd neighbors** command.

```
device# show bfd neighbors

Flags: * indicates State is inconsistent across the cluster
OurAddr     NeighAddr   State     Int
=======     =========   =====     ===
7.7.7.1     7.7.7.2     UP        Eth 0/1/2
```

This example output displays BFD neighbor information for the default VRF.

3. Enter the **show bfd neighbors application** command with the **ospf** and **details** keywords.

```
device# show bfd neighbors application ospf details

Flags: * indicates State is inconsistent across the cluster
OurAddr     NeighAddr   State     Int
=======     =========   =====     ===
7.7.7.1     7.7.7.2     UP        Eth 0/1/2

Local    State: UP  Diag: 0  Demand mode: 0 Poll: 0
Received State: UP  Diag: 0  Demand mode: 0 Poll: 0 Final: 0
Local    MinTxInt(ms): 1000   MinRxInt(ms): 1000   Multiplier: 5
Received MinTxInt(ms): 1000   MinRxInt(ms): 1000   Multiplier: 5
  Rx Count: 3806   Tx Count: 4308
  LD/RD: 10001/10001   Heard from Remote: Y
Current Registered Protocols: ospf
Uptime: 0 day 0 hour 0 min 0 sec 0 msec
```

This example displays detailed information about neighbor OSPF sessions.

4. Enter the **show bfd neighbors dest-ip** command with the **details** keyword.

```
device# show bfd neighbors dest-ip 1.1.1.6 details
```

This example shows detailed neighbor information about a destination device.

5. Enter the **show bfd neighbors interface** command with the **ve** and **details** keywords.

```
device# show bfd neighbors interface ve 5 details
```

This example shows detailed neighbor information about a specified VE interface.

6. Enter the **show bfd neighbors vrf** command and specify a VRF.

```
device# show bfd neighbors vrf default-vrf
```

This example shows BFD neighbor information for the default VRF.

7. Enter the **show ip bgp neighbors** command.

```
device# show ip bgp neighbors
```

This output shows BGP neighbor information including configured BFD information.

8. Enter the **show ip ospf** command.

```
device# show ip ospf
```

This output displays OSPF information including configured BFD information.

9. Enter the **show ip ospf neighbor** command with the **extensive** keyword.

```
device# show ip ospf neighbor extensive
```

This output displays detailed information about all neighbors including configured BFD information.

10. Enter the **show ipv6 ospf neighbor detail** command.

```
device# show ipv6 ospf neighbor detail
```

This output displays detailed OSPF neighbor information including configured BFD information.

Displaying BFD information

Various **show** commands verify information about BFD configurations.

Use one or more of the following commands to verify BFD information. The commands do not have to be entered in this order.

1. Enter the **show bfd** command.

```
device# show bfd

  BFD State: ENABLED, Version: 1
  Supported Protocols: bgp, static-ip, tunnel, ospf, ospf6
  All Sessions: Current: 72 Max Allowed: 250 Max Exceeded Count: 0

  Port            MinTx       MinRx       Mult Sessions
  ====            =====       =====       ==== ========
  Eth 0/51        500         500         3    1
  Eth 0/54        500         500         3    1
  Ve 3001         500         500         3    2
  Ve 3009         500         500         3    2
  Ve 3011         500         500         3    2
```

This example output displays BFD information.

2. Enter the **show bfd neighbors** command.

```
device# show bfd neighbors

OurAddr                                  NeighAddr                                State     Int
=======                                  =========                                =====     ===
118.113.0.0                              118.113.0.1                              DOWN      Eth 0/54
113.113.113.113                          118.118.118.118                          UP        Lo 1
```

This example output displays BFD neighbor information for the default VRF.

3. Enter the **show bfd neighbors application** command with the **ospf** and **details** keywords.

```
device# show bfd neighbors application ospf details

OurAddr                                  NeighAddr                                State     Int
=======                                  =========                                =====     ===
31.81.181.3                              31.81.181.8                              UP        Ve 3181

  Local    State: UP            Diag: 0              Demand mode: 0    Poll: 0
  Received State: UP            Diag: 0              Demand mode: 0    Poll: 0    Final: 0
  Local    MinTxInt(ms): 500    MinRxInt(ms): 500    Multiplier: 3
  Received MinTxInt(ms): 500    MinRxInt(ms): 500    Multiplier: 3
  Rx Count: 127097              Tx Count: 142393
  LD/RD:          3/81          Heard from Remote: Y
  Current Registered Protocols: ospf
  Uptime: 0 day 15 hour 53 min 12 sec 939 msec
```

This example displays detailed information about neighbor OSPF sessions.

4. Enter the **show bfd neighbors dest-ip** command with the **details** keyword.

```
device# show bfd neighbors dest-ip 118.118.118.118 details

OurAddr                                  NeighAddr                                State     Int
=======                                  =========                                =====     ===
113.113.113.113                          118.118.118.118                          UP        Lo 1

  Local    State: UP            Diag: 0              Demand mode: 0    Poll: 0
  Received State: UP            Diag: 0              Demand mode: 0    Poll: 0    Final: 0
  Local    MinTxInt(ms): 500    MinRxInt(ms): 500    Multiplier: 3
  Received MinTxInt(ms): 500    MinRxInt(ms): 500    Multiplier: 3
  Rx Count: 941487              Tx Count: 1016042
  LD/RD:          2/69          Heard from Remote: Y
  Current Registered Protocols: static-ip
  Uptime: 0 day 15 hour 54 min 20 sec 683 msec
```

This example shows detailed neighbor information about a destination device.

5. Enter the **show bfd neighbors interface** command with the **ve** and **details** keywords.

```
device# show bfd neighbors interface Ve 3009 details

OurAddr                                  NeighAddr                                State     Int
=======                                  =========                                =====     ===
30.9.1.3                                 30.9.1.8                                 UP        Ve 3009

  Local    State: UP            Diag: 0              Demand mode: 0    Poll: 0
  Received State: UP            Diag: 0              Demand mode: 0    Poll: 0    Final: 1
  Local    MinTxInt(ms): 500    MinRxInt(ms): 500    Multiplier: 3
  Received MinTxInt(ms): 500    MinRxInt(ms): 500    Multiplier: 3
  Rx Count: 147322              Tx Count: 142636
  LD/RD:          6/84          Heard from Remote: Y
  Current Registered Protocols: bgp
  Uptime: 0 day 15 hour 54 min 52 sec 989 msec
```

This example shows detailed neighbor information about a specified VE interface.

6. Enter the **show bfd neighbors vrf** command and specify a VRF.

```
device# show bfd neighbors vrf v3001

OurAddr                                  NeighAddr                                State     Int
=======                                  =========                                =====     ===
30.9.1.3                                 30.9.1.8                                 UP        Ve 3009
30.1.1.3                                 30.1.1.8                                 UP        Ve 3001
fdcd:3001:3009:1::113                    fdcd:3001:3009:1::118                    UP        Ve 3009
fe80::629c:9fff:fe5b:601                 fe80::629c:9fff:fe87:3601                UP        Ve 3001
```

This example shows BFD neighbor information for the default VRF.

7. Enter the **show ip ospf** command.

```
device# show ip ospf

OSPF Version                  Version 2
Router Id                     113.113.113.113
ASBR Status                   No
ABR Status                    No         (0)
Redistribute Ext Routes from
Initial SPF schedule delay    0          (msecs)
Minimum hold time for SPFs    0          (msecs)
Maximum hold time for SPFs    0          (msecs)
External LSA Counter          0
External LSA Checksum Sum     0
Originate New LSA Counter     2
Rx New LSA Counter            1
External LSA Limit            14913080
Administrative Distance
  -  External Routes:         110
  -  Intra Area Routes:       110
  -  Inter Area Routes:       110
  -  Route Map:               0
      -  Route Map Name:
      -  Source IP:
      -  Source IP Mask:
Database Overflow Interval    0
Database Overflow State :     NOT OVERFLOWED
RFC 1583 Compatibility :      Disabled
NSSA Translator:              Enabled

    None
```

This output displays OSPF information including configured BFD information.

8. Enter the **show ip ospf** command with the **extensive** keyword.

```
device# show ip ospf neighbor extensive

Number of Neighbors is 1, in FULL state 1

Port           Address         Pri State      Neigh Address   Neigh ID        Ev      Opt        Cnt
Ve 3011        30.11.11.3      1   FULL/DR    30.11.11.8      30.11.11.8      5       --O-----E- 0
  Neighbor is known for 0d:15h:57m:39s and up for 0d:15h:57m:39s
  Neighbor BFD State:UP        , HoldoverInterval(sec):Configured:0 Current:0
```

This output displays detailed information about all neighbors including configured BFD information.

9. Enter the **show ipv6 ospf neighbor detail** command.

```
device# show ipv6 ospf neighbor detail

Total number of neighbors in all states: 1
Number of neighbors in state Full      : 1

RouterID        Pri State    DR              BDR             Interface     [State]
30.11.11.8        1 Full     30.11.11.8      30.11.11.3      Ve 3011       [BDR]
                Option: 00-00-00    QCount: 0    Timer: 725
BFD State: UP, BFD HoldoverInterval(sec):Configured: 0 Current: 0
```

This output displays detailed OSPF neighbor information including configured BFD information.

Displaying BFD information

Various show commands can be used to display information about IKEv2 configurations.

Use one or more of the following commands to verify BFD information. The commands do not have to be entered in this order.

- Enter the **show bfd** command to display global BFD configuration and runtime information.

```
device# show bfd

BFD State: ENABLED, Version: 1
Current Registered Protocols: ospf
All Sessions: Current: 2 Max Allowed: 250 Max Exceeded Count: 0
Agent Sessions: Max Allowed on LC: 40 Max Exceeded Count for LCs: 0
BFD Enabled ports count: 6

Port      MinTx   MinRx   Mult  Sessions
eth 0/1   1000    1000    3     2
```

- Enter the **show bfd neighbors** command to display neighbor information for all BFD sesssions. The following example displays neighbor information for the default VRF.

```
device# show bfd neighbors

OurAddr       NeighAddr    State Int

10.0.6.1      10.0.6.2     Up    eth 0/1

10.2.7.2      10.2.7.1     Up    Ve10
```

- Enter the **show bfd neighbors application** command specifying the **details** option to display detailed neighbor session information for a specific application.

```
device# show bfd neighbors application ospf details

OurAddr     NeighAddr    State  Int
10.0.6.1    10.0.6.2     UP     Eth 0/1

Local    State: UP  Diag: 0  Demand mode: 0 Poll: 0
Received State: UP  Diag: 0  Demand mode: 0 Poll: 0 Final: 0
Local    MinTxInt(ms): 1000   MinRxInt(ms): 1000   Multiplier: 5
Received MinTxInt(ms): 1000   MinRxInt(ms): 1000   Multiplier: 5
  Rx Count: 3806   Tx Count: 4308
  LD/RD: 10001/10001   Heard from Remote: Y
Current Registered Protocols: ospf
Uptime: 0 day 0 hour 0 min 0 sec 0 msec
```

- Enter the **show bfd neighbors dest-ip** command specifying the **details** option to display detailed neighbor information for destination devices.

```
device# show bfd neighbors dest-ip 10.118.118.118 details

OurAddr                                  NeighAddr                                State     Int
=======                                  =========                                =====     ===
10.113.113.113                          10.118.118.118                            UP        Lo 1

  Local    State: UP            Diag: 0              Demand mode: 0    Poll: 0
  Received State: UP            Diag: 0              Demand mode: 0    Poll: 0    Final: 0
  Local    MinTxInt(ms): 500    MinRxInt(ms): 500    Multiplier: 3
  Received MinTxInt(ms): 500    MinRxInt(ms): 500    Multiplier: 3
  Rx Count: 941487              Tx Count: 1016042
  LD/RD:          2/69          Heard from Remote: Y
  Current Registered Protocols: ospf
  Uptime: 0 day 15 hour 54 min 20 sec 683 msec
```

- Enter the **show bfd neighbors interface** command specifying the **ve** and **details** options to display detailed BFD neighbor information for a specific interface. The following example displays detailed BFD neighbor information for a virtual Ethernet (VE) interface (3109).

```
device# show bfd neighbors interface ve 3109 details

OurAddr                                  NeighAddr                                State     Int
=======                                  =========                                =====     ===
310.9.101.3                               10.9.101.8                               UP        Ve 3109

  Local    State: UP            Diag: 0              Demand mode: 0    Poll: 0
  Received State: UP            Diag: 0              Demand mode: 0    Poll: 0    Final: 1
  Local    MinTxInt(ms): 500    MinRxInt(ms): 500    Multiplier: 3
  Received MinTxInt(ms): 500    MinRxInt(ms): 500    Multiplier: 3
  Rx Count: 347776              Tx Count: 389488
  LD/RD:        122/197         Heard from Remote: Y
  Current Registered Protocols: bgp
  Uptime: 1 day 19 hour 28 min 19 sec 509 msec
```

- Enter the **show bfd neighbors vrf** command specifying a VRF to display BFD neighbor information for a specific VRF instance.

```
device# show bfd neighbors vrf v3001

OurAddr                                  NeighAddr                                State     Int
=======                                  =========                                =====     ===
2001:db8:9fff:fe5b::601                  2001:db8:9fff:fe87::3601                UP        Ve 3001
10.1.1.3                                 10.1.1.8                                 UP        Ve 3001
10.9.1.3                                 10.9.1.8                                 UP        Ve 3009
2001:db8:3009:1::113                     2001:db8:3009:1::118                    UP        Ve 3009
```

- Enter the **show ip ospf** command to display OSPF information including configured BFD information.

```
device# show ip ospf

OSPF Version                  Version 2
Router Id                     10.113.113.113
ASBR Status                   No
ABR Status                    No         (0)
Redistribute Ext Routes from
Initial SPF schedule delay    0          (msecs)
Minimum hold time for SPFs    0          (msecs)
Maximum hold time for SPFs    0          (msecs)
External LSA Counter          0
External LSA Checksum Sum     0
Originate New LSA Counter     2
Rx New LSA Counter            1
External LSA Limit            14913080
Administrative Distance
  -  External Routes:         110
  -  Intra Area Routes:       110
  -  Inter Area Routes:       110
  -  Route Map:               0
      -  Route Map Name:
      -  Source IP:
      -  Source IP Mask:
Database Overflow Interval    0
Database Overflow State :     NOT OVERFLOWED
RFC 1583 Compatibility :      Disabled
NSSA Translator:              Enabled

    None
```

- Enter the **show ip ospf** command specifying the **extensive** option to display detailed OSPF information about all neighbors including configured BFD information.

```
device# show ip ospf neighbor extensive

Number of Neighbors is 1, in FULL state 1
```

```
Port           Address         Pri State      Neigh Address   Neigh ID        Ev      Opt        Cnt
Ve 3011        10.11.11.3      1   FULL/DR    10.11.11.8      10.11.11.8      5       --O-----E- 0
  Neighbor is known for 0d:15h:57m:39s and up for 0d:15h:57m:39s
  Neighbor BFD State:UP        , HoldoverInterval(sec):Configured:0 Current:0
```

- Enter the **show ipv6 ospf neighbor detail** command to display detailed OSPF neighbor information including configured BFD information.

```
device# show ipv6 ospf neighbor detail

Total number of neighbors in all states: 1
Number of neighbors in state Full      : 1

RouterID        Pri State    DR              BDR             Interface     [State]
10.11.11.8        1 Full     10.11.11.8      10.11.11.3      Ve 3011       [BDR]
                Option: 00-00-00    QCount: 0    Timer: 725
BFD State: UP, BFD HoldoverInterval(sec):Configured: 0 Current: 0
```

BGP4

- BGP4 overview....230
- BGP4 overview....234
- BGP4 peering....235
- BGP4 message types....235
- BGP4 attributes....238
- BGP4 best path selection algorithm....238
- BGP4 limitations and considerations....239
- Implementation of BGP4....239
- Device ID....240
- BGP global mode....240
- BGP global mode....241
- Configuring a local AS number....242
- IPv4 unicast address family....243
- IPv4 multicast address family....245
- Neighbor configuration....246
- Peer groups....250
- Advertising the default BGP4 route....251
- Four-byte AS numbers....252
- Cooperative BGP4 route filtering....253
- BGP4 parameters....254
- Route redistribution....255
- Advertised networks....256
- Importing routes into BGP4....257
- Static networks....258
- Route reflection....259
- Route flap dampening....261
- Aggregating routes advertised to BGP neighbors....261
- Advertising the default BGP4 route....262
- Advertising the default BGP4 route to a specific neighbor....263
- Multipath load sharing....264
- Specifying the weight added to received routes....264
- Using the IPv4 default route as a valid next hop for a BGP4 route....265
- Adjusting defaults to improve routing performance....266
- Next-hop recursion....267
- Route filtering....268
- BGP VRF route filters....269
- BGP regular expression pattern-matching characters....271
- Timers....272
- Enabling BGP4 in a non-default VRF....272
- BGP4 outbound route filtering....273
- BGP4 confederations....276
- BGP community and extended community....278
- BGP Large Communities....282
- BGP flowspec....285
- BGP4 graceful restart....300
- BGP dynamic neighbors....303
- BGP add path overview....305
- BGP additional-paths overview....310

- BGP best external overview....318
- BGP additional-paths....319
- BGP best-external route....328
- Auto shutdown of BGP neighbors on initial configuration....330
- BGP4 graceful shutdown....332
- BGP automatic neighbor discovery....337
- Generalized TTL Security Mechanism support....343
- Disabling the BGP AS_PATH check function....345
- Using route maps....345
- Matching on an AS-path....347
- Matching on a community ACL....347
- Matching on a destination network....348
- Matching on a BGP4 static network....349
- Matching on a next-hop device....351
- Matching on an interface....351
- Using route-map continue statements....352
- Route-map continue statement for BGP4 routes....353
- Using a route map to configure dampening....354
- Clearing diagnostic buffers....354
- Displaying BGP4 statistics....355
- Displaying BGP4 statistics....357
- Displaying BGP4 statistics....359
- Displaying BGP4 statistics....361
- Displaying BGP4 neighbor statistics....364
- Displaying BGP4 neighbor statistics....367
- Displaying BGP4 neighbor statistics....370
- Clearing BGP4 dampened paths....374
- Clearing BGP4 dampened paths....374
- BGP PIC....375

BGP4 overview

Border Gateway Protocol version 4 (BGP4) is an exterior gateway protocol that performs inter-autonomous system (AS) or inter-domain routing. It peers to other BGP-speaking systems over TCP to exchange network reachability and routing information. BGP primarily performs two types of routing: inter-AS routing, and intra-AS routing. BGP peers belonging to different autonomous systems use the inter-AS routing, referred as Exterior BGP (EBGP). On the other hand, within an AS BGP can be used to maintain a consistent view of network topology, to provide optimal routing, or to scale the network.

BGP is a path vector protocol and implements this scheme on large scales by treating each AS as a single point on the path to any given destination. For each route (destination), BGP maintains the AS path and uses this to detect and prevent loops between autonomous systems.

The Open Shortest Path First (OSPF) protocol (supported in Network OS 3.0 and later) provides dynamic routing within the VCS and internal domain. However, even though OSPF suffices for most of the routing needs within the VCS, an exterior gateway protocol such as BGP is needed for inter-domain routing outside the VCS domain.

Similar to other Layer 3 protocols in Network OS, BGP is supported only in the VCS mode of operation. Each RBridge in a VCS fabric running BGP acts as an individual BGP device. BGP can form iBGP peering with other RBridges in the same VCS fabric running BGP.

BGP support

Support for BGP on *Network OS* platforms is for BGP4 (compliant with RFC 1771 and 4271), and provides the following:

- Connectivity from the VCS to a core/external network or cloud
- A foundation to support virtual routing and forwarding (VRF) for multi-tenancy and remote-VCS access and route distribution across VRFs
- A foundation to support VRF scaling (OSPF does not scale well with lots of VRFs)
- A foundation to support OSPF Interior Gateway Protocol (IGP) scaling needs in future

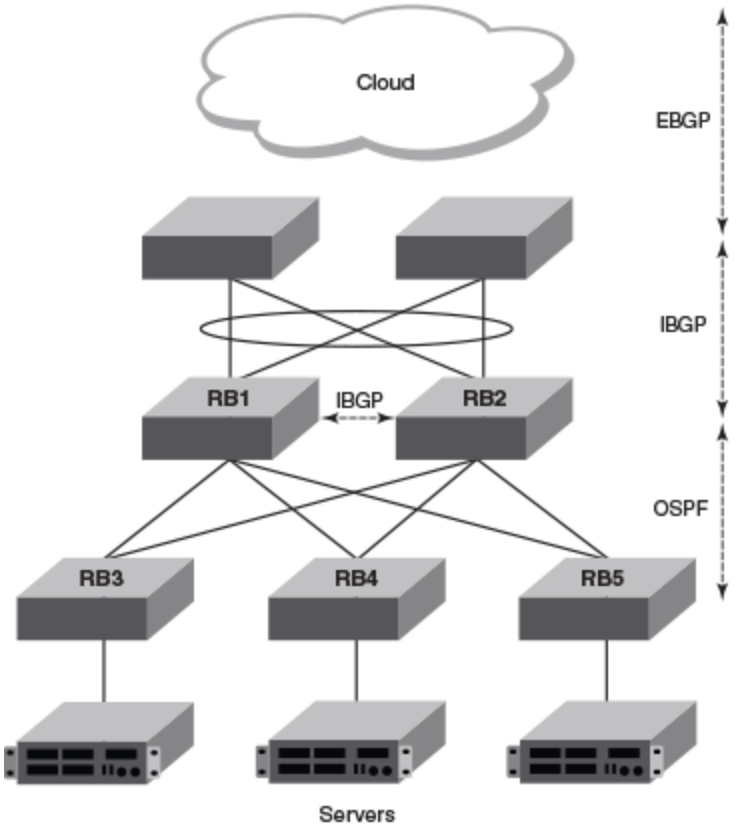
NOTE
An L3 License is required to enable BGP routing.

Deployment scenarios

BGP is typically used in a VCS Fabric at the aggregation layer and in connecting to the core. Routing bridges at the aggregation layer can either connect directly to the core, or connect through an additional device that aggregates traffic from the RBridges. The topologies below illustrate connectivity with and without an aggregator to the core. The details of these topologies are discussed in subsequent sections.

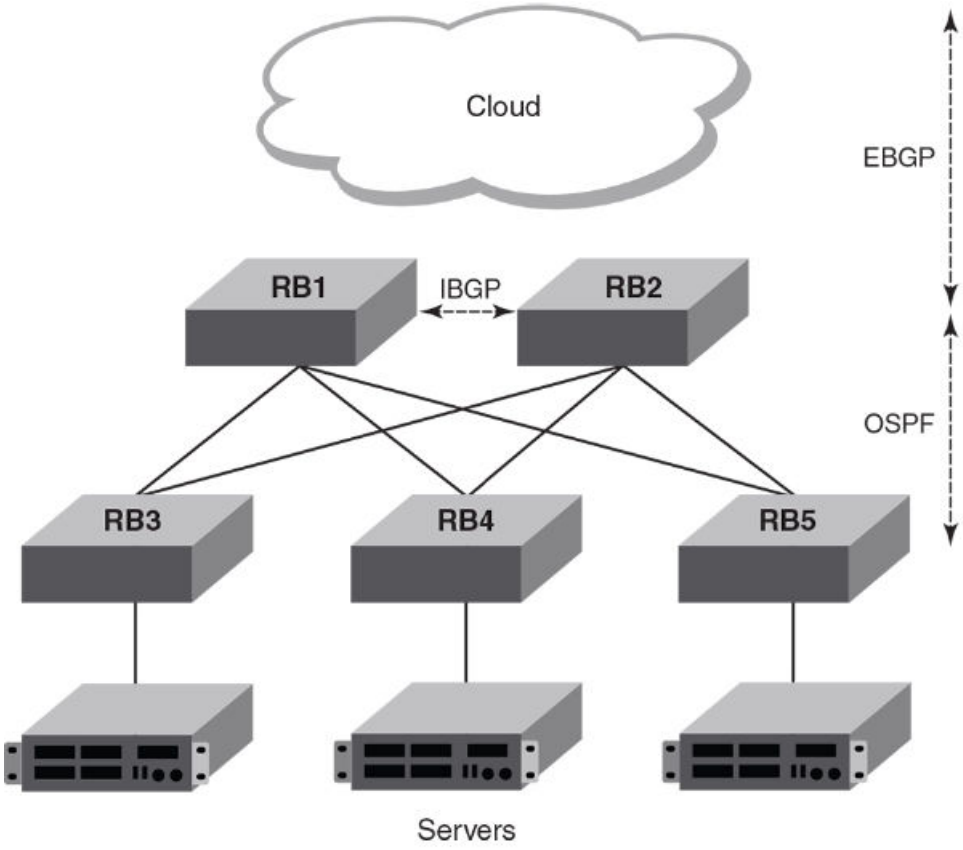
The figure below illustrates connectivity to the core through an aggregator. The RBridges use OSPF and IBGP to communicate with each other, connecting to the core aggregator through IBGP. The core aggregator connects in turn to the core through EBGP.

FIGURE 30 Connectivity to the core through an aggregator



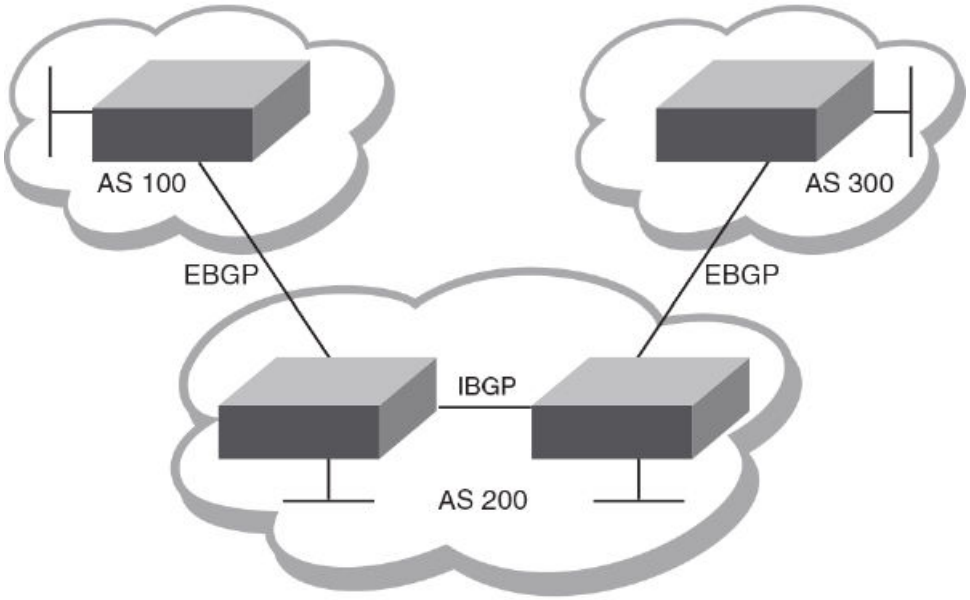
The figure below illustrates the previous topology but without an aggregator.

FIGURE 31 Connectivity to the core without an aggregator



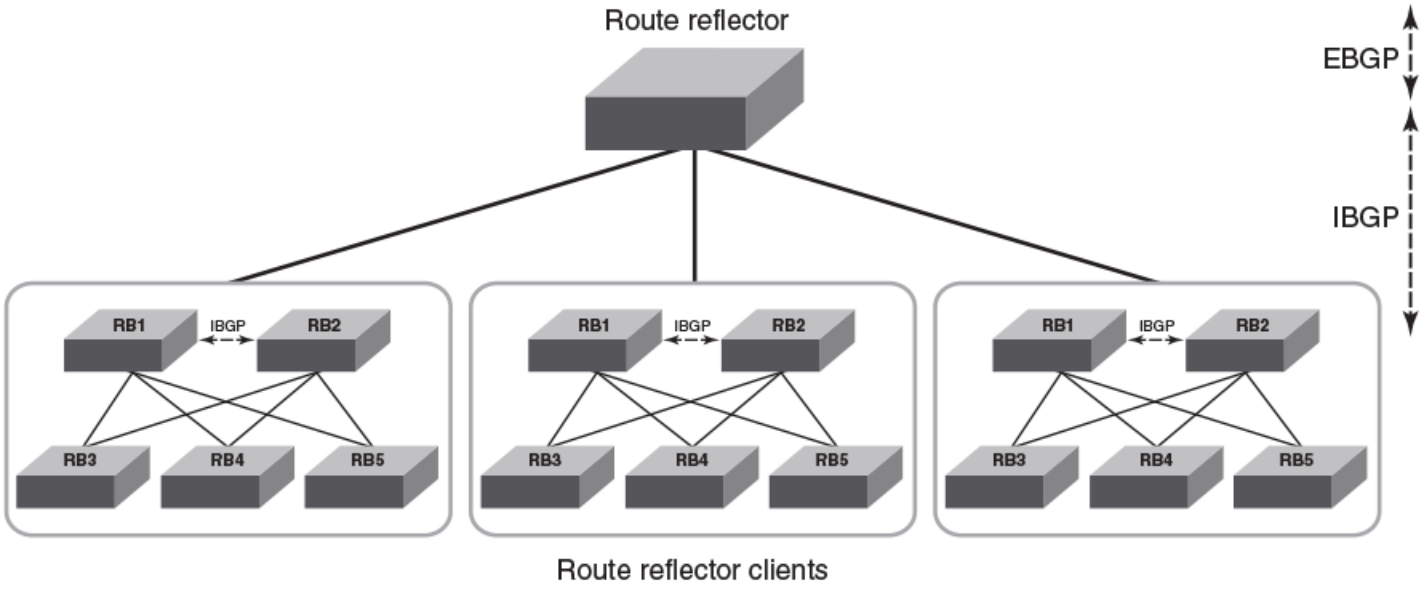
The figure below illustrates the role of BGP in communicating through multiple VCS clusters and autonomous systems.

FIGURE 32 BGP with multiple VCS clusters and autonomous systems



The figure below illustrates a BGP topology that incorporates a route-reflector server and route-reflector clients.

FIGURE 33 BGP route-reflector server and clients



BGP4 overview

Border Gateway Protocol version 4 (BGP4) is an exterior gateway protocol that performs inter-autonomous system (AS) or inter-domain routing. It peers to other BGP-speaking systems over TCP to exchange network reachability and routing information. BGP primarily performs two types of routing: inter-AS routing, and intra-AS routing. BGP peers belonging to different autonomous systems use the inter-AS routing, referred as Exterior BGP (eBGP). On the other hand, within an AS BGP can be used to maintain a consistent view of network topology, to provide optimal routing, or to scale the network.

BGP is a path vector protocol and implements this scheme on large scales by treating each AS as a single point on the path to any given destination. For each route (destination), BGP maintains the AS path and uses this to detect and prevent loops between autonomous systems.

Devices within an AS can use different Interior Gateway Protocols (IGPs) such as RIP, IS-IS, and OSPF to communicate with one another. However, for devices in different autonomous systems to communicate, they need to use an EGP. BGP4 is the standard EGP used by Internet devices and therefore is the EGP implemented on NetIron devices.

Devices within an AS can use different Interior Gateway Protocols (IGPs) such as IS-IS and OSPF to communicate with one another. However, for devices in different autonomous systems to communicate, they need to use an EGP. BGP4 is the standard EGP used by Internet devices and therefore is the EGP implemented on SLX-OS devices.

Devices within an AS can use different Interior Gateway Protocols (IGPs) such as OSPF to communicate with one another. However, for devices in different autonomous systems to communicate, they need to use an EGP. BGP4 is the standard EGP used by Internet devices and therefore is the EGP implemented on SLX-OS devices.

Devices within an AS can use different Interior Gateway Protocols (IGPs) such as OSPF to communicate with one another. However, for devices in different autonomous systems to communicate, they need to use an EGP. BGP4 is the standard EGP used by Internet devices and therefore is the EGP implemented on SLX-OS devices.

This is a simple example of two BGP4 ASs. Each AS contains three BGP4 devices. All of the BGP4 devices within an AS communicate using iBGP. BGP4 devices communicate with other autonomous systems using eBGP. Notice that each of the devices also is running an Interior Gateway Protocol (IGP). The devices in AS1 are running OSPF and the devices in AS2 are running RIP. The device can be configured to redistribute routes among BGP4, IS-IS, RIP, and OSPF. They also can redistribute static routes.

This is a simple example of two BGP4 ASs. Each AS contains three BGP4 devices. All of the BGP4 devices within an AS communicate using iBGP. BGP4 devices communicate with other autonomous systems using eBGP. Notice that each of the devices also is running an Interior Gateway Protocol (IGP). The devices in AS1 are running OSPF and the devices in AS2 are running IS-IS. The device can be configured to redistribute routes among BGP4, IS-IS, and OSPF. They also can redistribute static routes.

FIGURE 34 BGP4 autonomous systems

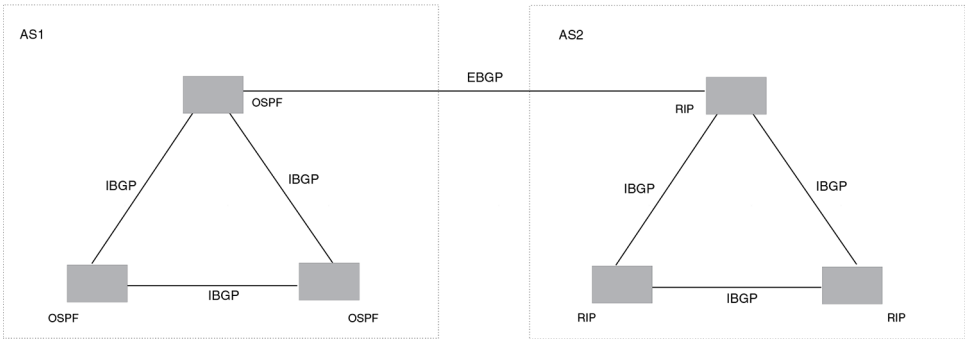
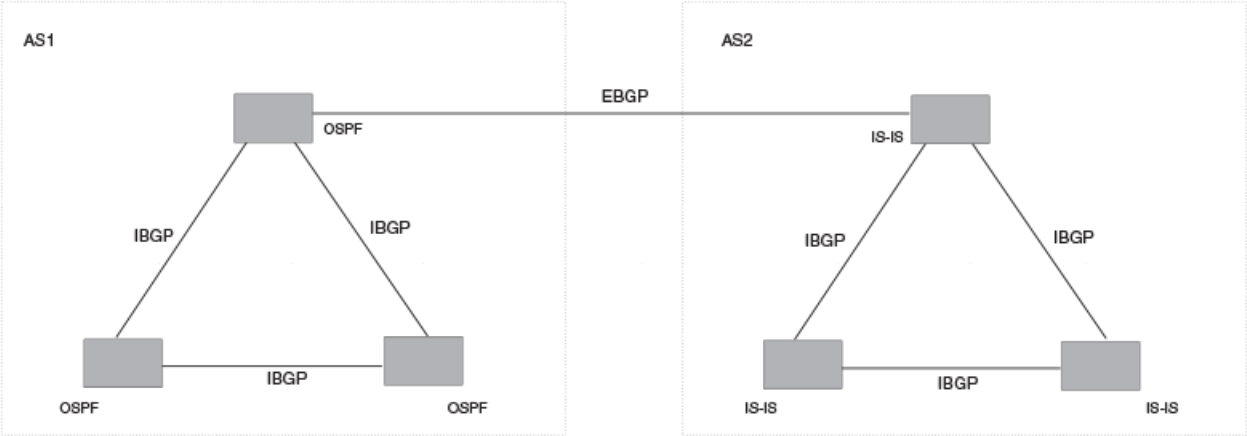


FIGURE 35 BGP4 autonomous systems



BGP4 peering

Unlike OSPF or other IGP protocols, BGP4 does not have neighbor detection capability. BGP4 neighbors (or peers) must be configured manually. A device configured to run BGP4 is called a BGP "speaker." A BGP speaker connects to another speaker (either in the same or a different AS) by using a TCP connection to port 179 (the well-known BGP port), to exchange the routing information. The TCP connection is maintained throughout the peering session. While the connection between BGP peers is alive, two peers communicate by means of the following types of messages:

- OPEN
- UPDATE
- KEEPALIVE
- NOTIFICATION
- ROUTE REFRESH

BGP4 peering can be internal or external, depending on whether the two BGP peers belong to the same AS or different ASs. A BGP4 session between peers within a single AS is referred to as an Interior BGP (iBGP) session; a session between peers belonging to different ASs is referred to as an Exterior BGP (eBGP) session.

In order to establish a TCP connection between two iBGP peers, the IP reachability should be established either by means of the underlying IGP protocol (e.g. OSPF) or by means of static routes. When routes are advertised within iBGP peers, the following primary actions are taken in contrast to eBGP peering:

- Routes learned from an iBGP peer are not usually advertised to other iBGP peers, in order to prevent loops within an AS.
- Path attributes are not usually changed, in order to maintain the best path selection at other nodes within an AS.
- The AS path and next hop are not normally changed.

BGP4 message types

All BGP4 messages use a common packet header, with the following byte lengths:

Marker	Length	Type	Data
16	2	1	variable

NOTE
All values in the following tables are in bytes.

Type can be OPEN, UPDATE, NOTIFICATION, KEEPALIVE, or ROUTE-REFRESH, as described below.

OPEN message

After establishing TCP connection, BGP peers exchange OPEN message to identify each other.

Version	Autonomous System	Hold-Time	BGP Identifier	Optional Parameter Len	Optional Parameters
1	2 or 4	2	4	1	4

Version

Only BGP4 version 4 is supported.

Autonomous System

Both 2-byte and 4-byte AS numbers are supported.

KEEPALIVE and **HOLDTIME** messages

A BGP **timer** command specifies both **keep-alive** and **hold-time** operands that manage the intervals for BGP KEEPALIVE and HOLDTIME messages. The keep alive time specifies how frequently the device sends KEEPALIVE messages to its BGP4 neighbors. The hold time specifies how long the device waits for a KEEPALIVE or UPDATE message from a neighbor before concluding that the neighbor is dead. When two neighbors have different hold-time values, the lowest value is used. A hold-time value of 0 means "always consider neighbor to be active."

Refer to the *Extreme NetIron Command Reference* for more information.

Refer to the *Extreme Network OS Command Reference* for more information.

Refer to the *Extreme SLX-OS Command Reference* for more information.

BGP Identifier

Indicates the router (or device) ID of the sender. When router-id is not configured, device-id is taken from the loopback interface. Otherwise, the lowest IP address in the system is used.

Parameter List

Optional list of additional parameters used in peer negotiation.

UPDATE message

The UPDATE message is used to advertise new routes, withdraw previously advertised routes, or both.

WithdrawnRoutesLength	WithdrawnRoutes	Total PathAttributes Len	Path Attributes	NLRI
2	variable	2	variable	variable

Withdrawn Routes Length

Indicates the length of next (withdrawn routes) field. It can be 0.

Withdrawn Routes

Contains list of routes (or IP-prefix/Length) to indicate routes being withdrawn.

Total Path Attribute Len

Indicates length of next (path attributes) field. It can be 0.

Path Attributes

Indicates characteristics of the advertised path. Possible attributes: Origin, AS Path, Next Hop, MED (Multi-Exit Discriminator), Local Preference, Atomic Aggregate, Aggregator, Community, extended-Communities.

NLRI

Network Layer Reachability Information — the set of destinations whose addresses are represented by one prefix. This field contains a list of IP address prefixes for the advertised routes.

NOTIFICATION message

In case of an error that causes the TCP connection to close, the closing peer sends a notification message to indicate the type of error.

Error Code	ErrorSubcode	Error Data
1	1	variable

Error Code

Indicates the type of error, which can be one of following:

- Message header error
- Open message error
- Update message error
- Hold timer expired
- Finite state-machine error
- Cease (voluntarily)

Error Subcode

Provides specific information about the error reported.

Error Data

Contains data based on error code and subcode.

KEEPALIVE message

Because BGP does not regularly exchanges route updates to maintain a session, KEEPALIVE messages are sent to keep the session alive. A KEEPALIVE message contains just the BGP header without data field. Default KEEPALIVE time is 60 seconds and is configurable.

REFRESH message

A REFRESH message is sent to a neighbor requesting that the neighbor resend the route updates. This is useful when the inbound policy has been changed.

BGP4 attributes

BGP4 attributes are passed in UPDATE messages to describe the characteristics of a BGP path by the advertising device. At a high level, there are only two types of attributes: well-known and optional. All of the well-known attributes, as described in RFC 4271, are supported.

BGP4 best path selection algorithm

The BGP decision process is applied to the routes contained in the Routing Information Base, Incoming (RIB-In) which contains routes learned from inbound update messages. The output of the decision process is the set of routes that will be advertised to BGP speakers in local or remote autonomous systems and are stored in the Adjacency RIB, Outgoing (RIB-Out).

When multiple paths for the same route prefix are known to a BGP4 device, the device uses the following algorithm to weigh the paths and determine the optimal path for the route. The optimal path depends on various parameters, which can be modified.

Refer to the *Extreme NetIron Command Reference* for more information.

Refer to the *Extreme Network OS Command Reference* for more information.

Refer to the *Extreme SLX-OS Command Reference* for more information.

1. Verify that the next hop can be resolved by means of Interior Gateway Protocol (IGP).
2. Use the path with the largest weight.
3. If the weights are the same, prefer the path with the largest local preference.
4. Prefer the route that was self-originated locally.
5. If the local preferences are the same, prefer the path with the shortest AS-path. An AS-SET counts as 1. A confederation path length, if present, is not counted as part of the path length.

   The **as-path ignore** command disables the comparison of the AS path lengths of otherwise equal paths.

   NOTE
   This step can be skipped if the **as-path-ignore** command is configured.

6. If the AS-path lengths are the same, prefer the path with the lowest origin type. From low to high, route origin types are valued as follows:
   - IGP is lowest.
   - EGP is higher than IGP but lower than INCOMPLETE.
   - INCOMPLETE is highest.
7. If the paths have the same origin type, prefer the path with the lowest MED.

   The device compares the MEDs of two otherwise equivalent paths if and only if the routes were learned from the same neighboring AS. This behavior is called deterministic MED. Deterministic MED is always enabled and cannot be disabled.

   To ensure that the MEDs are always compared, regardless of the AS information in the paths, the **always-compare-med** command can be used. This option is disabled by default.

   The **med-missing-as-worst** command can be used to make the device regard a BGP4 route with a missing MED attribute as the least-favorable path when the MEDs of the route paths are compared.

   MED comparison is not performed for internal routes that originate within the local AS or confederation, unless the **compare-med-empty-aspath** command is configured.

8. Prefer paths in the following order:
   - Routes received through eBGP from a BGP4 neighbor outside of the confederation
   - Routes received through eBGP from a BGP4 device within the confederation *or* routes received through IBGP.
9. If all the comparisons above are equal, prefer the route with the lowest IGP metric to the BGP4 next hop. This is the closest internal path inside the AS to reach the destination.
10. If the internal paths also are the same and BGP4 load sharing is enabled, load-share among the paths. Otherwise go to Step 11.

    NOTE
    For eBGP routes, load sharing applies only when the paths are from neighbors within the same remote AS. eBGP paths from neighbors in different ASs are not compared, unless multipath multi-as is enabled.

11. If **compare-routerid** is enabled, prefer the path that comes from the BGP4 device with the lowest device ID. If a path contains originator ID attributes, then the originator ID is substituted for the router ID in the decision.
12. Prefer the path with the minimum cluster-list length.
13. Prefer the route that comes from the lowest BGP4 neighbor address.

BGP4 limitations and considerations

The following limitations and considerations apply to BGP4 support on Network OS platforms:

- There is no backward compatibility. In case of a downgrade, BGP configurations are lost.
- There is no support for nonstop routing.
- RASLogs are generated when a BGP session begins.
- RASTRACE logs are available with module ID "261 BGP".
- BGP logs are generated with module ID "BGP-1002".

Implementation of BGP4

BGP4 is described in RFC 1771 and the latest BGP4 drafts. The Extreme BGP4 implementation fully complies with RFC 1771. Extreme BGP4 implementation also supports the following RFCs:

- RFC 1745 (OSPF Interactions)
- RFC 1997 (BGP Communities Attributes)
- RFC 2385 (TCP MD5 Signature Option)
- RFC 2439 (Route Flap Dampening)
- RFC 2796 (Route Reflection)
- RFC 2842 (Capability Advertisement)
- RFC 3065 (BGP4 Confederations)
- RFC 2858 (Multiprotocol Extensions)
- RFC 2918 (Route Refresh Capability)
- RFC 3392 (BGP4 Capability Advertisement)
- Draft-ietf-idr-restart-10.txt (restart mechanism for BGP4)

Device ID

BGP automatically calculates the device identifier it uses to specify the origin in routes it advertises. If a router-id configuration is already present in the system, then device-id is used as the router-id. Otherwise, the device first checks for a loopback interface, and the IP address configured on that interface is chosen as the device-id. However, if a loopback interface is not configured, the device-id is chosen from lowest-numbered IP interface address configured on the device. Once device-id is chosen, the device identifier is not calculated unless the IP address configured above is deleted.

BGP global mode

To enable BGP4 use the **router bgp** command in RBridge ID configuration mode.

```
device(config-rbridge-id-12)# router bgp
```

To enable BGP4, use the **router bgp** command in global configuration mode.

```
device(config)# router bgp
```

After using the **router bgp** command you enter into BGP global configuration mode.

Configurations that are not specific to address-family configuration are available in the BGP global configuration mode:

```
device(config-bgp-router)# ?
Possible completions:
  address-family                Enter Address Family command mode
  always-compare-med            Allow comparing MED from different neighbors
  as-path-ignore                Ignore AS_PATH length for best route selection
  auto-shutdown-new-neighbors   Auto shutdown new neighbor
  bfd                           Set BFD global parameters for BGP
  capability                    Set capability
  cluster-id                    Configure Route-Reflector Cluster-ID
  compare-med-empty-aspath      Allow comparing MED from different neighbors
                                even with empty as-path attribute
  compare-routerid              Compare router-id for identical BGP paths
  confederation                 Configure AS confederation parameters
  default-local-preference      Configure default local preference value
  distance                      Define an administrative distance
  enforce-first-as              Enforce the first AS for EBGP routes
  fast-external-fallover        Reset session if link to EBGP peer goes down
  install-igp-cost              Install igp cost to nexthop instead of MED
                                value as BGP route cost
  local-as                      Configure local AS number
  log-dampening-debug           Log dampening debug messages
  maxas-limit                   Impose limit on number of ASes in AS-PATH attribute
  med-missing-as-worst          Consider routes missing MED attribute as least desirable
  neighbor                      Specify a neighbor router
  timers                        Adjust routing timers

device(config-bgp-router)# ?
Possible completions:
  address-family                Enter Address Family command mode
  always-compare-med            Allow comparing MED from different neighbors
  as-path-ignore                Ignore AS_PATH length for best route selection
  auto-shutdown-new-neighbors   Auto shutdown new neighbor
  capability                    Set capability
  cluster-id                    Configure Route-Reflector Cluster-ID
  compare-med-empty-aspath      Allow comparing MED from different neighbors
                                even with empty as-path attribute
  compare-routerid              Compare router-id for identical BGP paths
  confederation                 Configure AS confederation parameters
  default-local-preference      Configure default local preference value
  describe                      Display transparent command information
  distance                      Define an administrative distance
  enforce-first-as              Enforce the first AS for EBGP routes
```

```
  fast-external-fallover        Reset session if link to EBGP peer goes down
  install-igp-cost              Install igp cost to nexthop instead of MED
                                value as BGP route cost
  local-as                      Configure local AS number
  log-dampening-debug           Log dampening debug messages
  maxas-limit                   Impose limit on number of ASes in AS-PATH
                                attribute
  med-missing-as-worst          Consider routes missing MED attribute asleast
                                desirable
  neighbor                      Specify a neighbor router

  timers                        Adjust routing timers


device(config-bgp-router)# ?
Possible completions:
  address-family                Enter Address Family command mode
  always-compare-med            Allow comparing MED from different neighbors
  as-path-ignore                Ignore AS_PATH length for best route selection
  auto-shutdown-new-neighbors   Auto shutdown new neighbor
  bfd                           Set BFD global parameters for BGP
  capability                    Set capability
  cluster-id                    Configure Route-Reflector Cluster-ID
  compare-med-empty-aspath      Allow comparing MED from different neighbors
                                even with empty as-path attribute
  compare-routerid              Compare router-id for identical BGP paths
  confederation                 Configure AS confederation parameters
  default-local-preference      Configure default local preference value
  distance                      Define an administrative distance
  enforce-first-as              Enforce the first AS for EBGP routes
  fast-external-fallover        Reset session if link to EBGP peer goes down
  install-igp-cost              Install igp cost to nexthop instead of MED
                                value as BGP route cost
  local-as                      Configure local AS number
  log-dampening-debug           Log dampening debug messages
  maxas-limit                   Impose limit on number of ASes in AS-PATH attribute
  med-missing-as-worst          Consider routes missing MED attribute as least desirable
  neighbor                      Specify a neighbor router
  timers                        Adjust routing timers
```

BGP global mode

To enable BGP4, use the **router bgp** command in global configuration mode.

```
device(config)# router bgp
```

After using the **router bgp** command you enter into BGP global configuration mode.

Commands entered in BGP global configuration mode apply to the IPv4 unicast address family. Where relevant, this chapter discusses and provides IPv4-unicast-specific examples. You must first configure IPv4 unicast routing for any IPv4 routing protocol to be active.

```
Possible completions:

  additional-paths              Specify BGP  additional paths
  address-family                Enter Address Family command mode
  advertise-best-external       Advertise best external path to internal peers
  aggregate-address             Configure BGP aggregate entries
  always-compare-med            Allow comparing MED from different neighbors
  always-propagate              Allow readvertisement of best BGP routes not
                                in IP forwarding table
  as-path-ignore                Ignore AS_PATH length info for best route
                                selection
  auto-shutdown-new-neighbors   Automatically set new BGP neighbors to
                                shutdown state
  bfd                           Set BFD global parameters for BGP4
  bfd-enable                    Enable BFD for BGP4
  bgp-redistribute-internal     Allow redistribution of iBGP routes into IGPs
```

```
capability                      Set capability
clear                           Clear table/statistics/keys
client-to-client-reflection     Configure client to client route reflection
cluster-id                      Configure Route-Reflector Cluster-ID
compare-med-empty-aspath        Allow comparing MED from different neighbors
                                even with empty as-path attribute
compare-routerid                Compare router-id for identical BGP paths
confederation                   Configure AS confederation parameters
dampening                       Enable route-flap dampening
default-information-originat    Distribute a default route
default-local-preference        Configure default local preference value
default-metric                  Set metric of redistributed routes
distance                        Define an administrative distance
enforce-first-as                Enforce the first AS for EBGP routes
fast-external-fallover          Reset session if link to EBGP peer goes down
graceful-restart                Enables the BGP graceful restart capability
install-igp-cost                Install igp cost to nexthop instead of MED
                                value as BGP route cost
local-as                        Configure local AS number
log-dampening-debug             Log dampening debug messages
maxas-limit                     Impose limit on number of ASes in AS-PATH
                                attribute
maximum-paths                   Forward packets over multiple paths
med-missing-as-worst            Consider routes missing MED attribute as least
                                desirable
multipath                       Enable multipath for ibgp or ebgp neighbors
                                only
neighbor                        Specify a neighbor router
network                         Specify a network to announce via BGP
next-hop-enable-default         Enable default route for BGP next-hop lookup
next-hop-mpls                   Include MPLS routes for BGP next-hop lookup
next-hop-recursion              Perform next-hop recursive lookup for BGP route
redistribute                    Redistribute information from another routing
                                protocol
rib-route-limit                 Limit BGP rib count in routing table
show                            Display system information
static-network                  Special network that do not depends on IGP and
                                always treat as best route in BGP
table-map                       Map external entry attributes into routing
                                table
timers                          Adjust routing timers
update-time                     Configure igp route update interval
```

Configuring a local AS number

The local AS number (ASN) identifies the AS in which the BGP device resides. The following task configures the local ASN in which the device resides.

NOTE

Use well-known private ASNs in the range from 64512 through 65535 if the AS number of the organization is not known.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

```
device(config-bgp)# local-as 1000
```

The following example configures the local ASN for a device.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000

device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1000

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
```

IPv4 unicast address family

The IPv4 unicast address family configuration level provides access to commands that allow you to configure BGP4 unicast routes. The commands that you enter at this level apply only to the IPv4 unicast address family.

BGP4 supports the IPv4 address family configuration level. This configuration is applied in the IPv4 address-family unicast submode of BGP.

```
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)#
```

The commands that you can access while at the IPv4 unicast address family configuration level are also available at the IPv6 unicast address family configuration levels. Each address family configuration level allows you to access commands that apply to that particular address family only.

Where relevant, this chapter discusses and provides IPv4-unicast-specific examples.

The following configuration options are allowed under BGP IPv4 address family unicast configuration mode:

```
device(config-bgp-router)# address-family ipv4 unicast

device(config-bgp-ipv4u)# ?
Possible completions:
  aggregate-address               Configure BGP aggregate entries
  always-propagate                Allow readvertisement of best BGP routes not
                                  in IP Forwarding table
  bgp-redistribute-internal       Allow redistribution of iBGP routes into IGPs
  client-to-client-reflection     Configure client to client route reflection
  dampening                       Enable route-flap dampening
  default-information-originate   Originate Default Information
  default-metric                  Set metric of redistributed routes
  graceful-restart                Enables the BGP graceful restart capability
  maximum-paths                   Forward packets over multiple paths
  multipath                       Enable multipath for ibgp or ebgp neighbors
                                  only
  neighbor                        Specify a neighbor router
  network                         Specify a network to announce via BGP
```

```
  next-hop-enable-default         Enable default route for BGP next-hop lookup
  next-hop-recursion              Perform next-hop recursive lookup for BGP
                                  route
  redistribute                    Redistribute information from another
                                  routing protocol
  rib-route-limit                 Limit BGP rib count in routing table
  static-network                  Special network that do not depends on IGP
                                  and always treat as best route in BGP
  table-map                       Map external entry attributes into routing
                                  table
  update-time                     Configure igp route update interval

device(config-bgp-router)# address-family ipv4 unicast

device(config-bgp-ipv4u)# ?
Possible completions:
  aggregate-address               Configure BGP aggregate entries
  always-propagate                Allow readvertisement of best BGP routes not
                                  in IP Forwarding table
  bgp-redistribute-internal       Allow redistribution of iBGP routes into IGPs
  client-to-client-reflection     Configure client to client route reflection
  dampening                       Enable route-flap dampening
  default-information-originate   Originate Default Information
  default-metric                  Set metric of redistributed routes
  graceful-restart                Enables the BGP graceful restart capability
  maximum-paths                   Forward packets over multiple paths
  multipath                       Enable multipath for ibgp or ebgp neighbors
                                  only
  neighbor                        Specify a neighbor router
  network                         Specify a network to announce via BGP
  next-hop-enable-default         Enable default route for BGP next-hop lookup
  next-hop-mpls                   Changes for IPoMPLS route download, pkt path.
  next-hop-recursion              Perform next-hop recursive lookup for BGP
                                  route
  redistribute                    Redistribute information from another
                                  routing protocol
  rib-route-limit                 Limit BGP rib count in routing table
  static-network                  Special network that do not depends on IGP
                                  and always treat as best route in BGP
  table-map                       Map external entry attributes into routing
                                  table
  update-time                     Configure igp route update interval

device(config-bgp-router)# address-family ipv4 unicast

device(config-bgp-ipv4u)# ?
Possible completions:
  aggregate-address               Configure BGP aggregate entries
  always-propagate                Allow readvertisement of best BGP routes not
                                  in IP Forwarding table
  bgp-redistribute-internal       Allow redistribution of iBGP routes into IGPs
  client-to-client-reflection     Configure client to client route reflection
  dampening                       Enable route-flap dampening
  default-information-originate   Originate Default Information
  default-metric                  Set metric of redistributed routes
  graceful-restart                Enables the BGP graceful restart capability
  maximum-paths                   Forward packets over multiple paths
  multipath                       Enable multipath for ibgp or ebgp neighbors
                                  only
  neighbor                        Specify a neighbor router
  network                         Specify a network to announce via BGP
  next-hop-enable-default         Enable default route for BGP next-hop lookup
  next-hop-recursion              Perform next-hop recursive lookup for BGP
                                  route
  redistribute                    Redistribute information from another
                                  routing protocol
  rib-route-limit                 Limit BGP rib count in routing table
  static-network                  Special network that do not depends on IGP
                                  and always treat as best route in BGP
  table-map                       Map external entry attributes into routing
```

```
                                  table
  update-time                     Configure igp route update interval

device(config-bgp-router)# address-family ipv4 unicast

device(config-bgp-ipv4u)# ?
Possible completions:
  aggregate-address               Configure BGP aggregate entries
  always-propagate                Allow readvertisement of best BGP routes not
                                  in IP Forwarding table
  bgp-redistribute-internal       Allow redistribution of iBGP routes into IGPs
  client-to-client-reflection     Configure client to client route reflection
  dampening                       Enable route-flap dampening
  default-information-originate   Originate Default Information
  default-metric                  Set metric of redistributed routes
  graceful-restart                Enables the BGP graceful restart capability
  maximum-paths                   Forward packets over multiple paths
  multipath                       Enable multipath for ibgp or ebgp neighbors
                                  only
  neighbor                        Specify a neighbor router
  network                         Specify a network to announce via BGP
  next-hop-enable-default         Enable default route for BGP next-hop lookup
  next-hop-recursion              Perform next-hop recursive lookup for BGP
                                  route
  redistribute                    Redistribute information from another
                                  routing protocol
  rib-route-limit                 Limit BGP rib count in routing table
  static-network                  Special network that do not depends on IGP
                                  and always treat as best route in BGP
  table-map                       Map external entry attributes into routing
                                  table
  update-time                     Configure igp route update interval
```

IPv4 multicast address family

The BGP4 multicast address family configuration level provides access to commands that allow you to configure BGP4 multicast routes. The commands that you enter at this level apply only to the IPv4 multicast address family.

BGP4 supports the IPv4 address family configuration level.

You can generate a configuration for BGP4 multicast routes that is separate and distinct from configurations for BGP unicast routes and IPv6 BGP multicast routes.

The commands that you can access while at the IPv4 multicast address family configuration level are also available at the IPv6 multicast address family configuration levels. Each address family configuration level allows you to access commands that apply to that particular address family only.

The following configurations are allowed under BGP IPv4 address-family multicast mode:

```
device(config-bgp-ipv4m)# ?

Possible completions:

  additional-paths               Specify BGP  additional paths
  address-family                 Enter Address Family command mode
  advertise-best-external        Advertise best external path to internal peers
  aggregate-address              Configure BGP aggregate entries
  always-propagate               Allow readvertisement of best BGP routes not
                                 in IP forwarding table
  bfd                            Set BFD global parameters for BGP4
  bfd-enable                     Enable BFD for BGP4
  clear                          Clear table/statistics/keys
  client-to-client-reflection    Configure client to client route reflection
  dampening                      Enable route-flap dampening
  default-information-originate  Distribute a default route
  default-metric                 Set metric of redistributed routes
```

```
  log-dampening-debug            Log dampening debug messages
  maximum-paths                  Forward packets over multiple paths
  multipath                      Enable multipath for ibgp or ebgp neighbors
                                 only
  neighbor                       Specify a neighbor router
  network                        Specify a network to announce via BGP
  next-hop-enable-default        Enable default route for BGP next-hop lookup
  redistribute                   Redistribute information from another routing
                                 protocol
  rib-route-limit                Limit BGP rib count in routing table
  show                           Display system information
  table-map                      Map external entry attributes into routing
                                 table
  update-time                    Configure igp route update interval
```

Neighbor configuration

For each neighbor a device is going to peer with, there must be a neighbor configuration that specifies an IP address (which must be the primary IP address of interface connection to get established) and an AS number of the neighbor. For each neighbor, you can specify a set of attributes. However, in cases where a set of neighbors share the same set of attributes, it is advisable to create a peer-group.

The neighbor configuration appears in both the global and address-family modes of BGP. The neighbor parameters that are common to all of the address families appear in BGP global configuration mode, while the parameters that are specific to an address family appear within the BGP address-family submode.

The following **running-config** excerpt illustrates the neighbor configurations that are allowed in BGP global mode and IPv4 address-family submode:

```
router bgp
  local-as 6500
  neighbor 10.231.64.10 advertisement-interval 60
  neighbor as-override
  neighbor 10.231.64.10 capability as4-enable
  neighbor 10.231.64.10 description "Example Neighbor Configuration"
  neighbor 10.231.64.10 ebgp-multihop 2
  neighbor 10.231.64.10 enforce-first-as
  neighbor 10.231.64.10 local-as 64900
  neighbor 10.231.64.10 maxas-limit in disable
  neighbor 10.231.64.10 next-hop-self always
  neighbor 10.231.64.10 password default
  neighbor 10.231.64.10 remote-as 1200
  neighbor 10.231.64.10 remove-private-as
  neighbor 10.231.64.10 shutdown generate-rib-out
  neighbor 10.231.64.10 soft-reconfiguration inbound
  neighbor 10.231.64.10 timers keep-alive 120 hold-time 240
  neighbor 10.231.64.10 update-source loopback lo0
  address-family ipv4 unicast
   neighbor 10.231.64.10 default-originate route-map test-map
   neighbor 10.231.64.10 filter-list [ 1 ] in
   neighbor 10.231.64.10 maximum-prefix 15000 teardown
   neighbor 10.231.64.10 prefix-list test-prefix in
   neighbor 10.231.64.10 route-map in test-map
   neighbor 10.231.64.10 send-community both
   neighbor 10.231.64.10 unsuppress-map test-map-2
   neighbor 10.231.64.10 weight 10
```

The following neighbor configuration options are allowed under BGP global configuration mode:

```
device(config-bgp-router)# neighbor 10.1.1.1 ?
Possible completions:
  advertisement-interval   Minimum interval between sending BGP routing updates
  as-override              Override matching AS-number while sending update
  capability               Advertise capability to the peer
```

```
  description              Neighbor by description
  ebgp-btsh                Enable EBGP TTL Security Hack Protection
  ebgp-multihop            Allow EBGP neighbors not on directly connected
                           networks
  enforce-first-as         Enforce the first AS for EBGP routes
  local-as                 Assign local-as number to neighbor
  maxas-limit              Impose limit on number of ASes in AS-PATH attribute
  next-hop-self            Disable the next hop calculation for this neighbor
  password                 Enable TCP-MD5 password protection
  peer-group               Assign peer-group to neighbor
  remote-as                Specify a BGP neighbor
  remove-private-as        Remove private AS number from outbound updates
  shutdown                 Administratively shut down this neighbor
  soft-reconfiguration     Per neighbor soft reconfiguration
  static-network-edge      Neighbor as special service edge, static-network
                           shall not be advertised if installed as DROP
  timers                   BGP per neighbor timers
  update-source            Source of routing updates

device(config-bgp-router)# neighbor 10.1.1.1 ?
Possible completions:
  advertisement-interval   Minimum interval between sending BGP routing updates
  as-override              Override matching AS-number while sending update
  bfd                      Enable BFD session for the neighbor
  capability               Advertise capability to the peer
  description              Neighbor by description
  ebgp-btsh                Enable EBGP TTL Security Hack Protection
  ebgp-multihop            Allow EBGP neighbors not on directly connected
                           networks
  enforce-first-as         Enforce the first AS for EBGP routes
  local-as                 Assign local-as number to neighbor
  maxas-limit              Impose limit on number of ASes in AS-PATH attribute
  next-hop-self            Disable the next hop calculation for this neighbor
  password                 Enable TCP-MD5 password protection
  peer-group               Assign peer-group to neighbor
  remote-as                Specify a BGP neighbor
  remove-private-as        Remove private AS number from outbound updates
  shutdown                 Administratively shut down this neighbor
  soft-reconfiguration     Per neighbor soft reconfiguration
  static-network-edge      Neighbor as special service edge, static-network
                           shall not be advertised if installed as DROP
  timers                   BGP per neighbor timers
  update-source            Source of routing updates
```

The following neighbor configuration options are allowed under BGP IPv4 address family unicast configuration mode:

```
device(config-bgp-ipv4u)# neighbor 10.1.1.1 ?
Possible completions:
  activate                 Allow exchange of route in the current family mode
  allowas-in               Disables the AS_PATH check of the routes learned
                           from the AS
  capability               Advertise capability to the peer
  default-originate        Originate default route to peer
  filter-list              Establish BGP filters
  maximum-prefix           Maximum number of prefix accept from this peer
  prefix-list              Prefix List for filtering routes
  route-map                Apply route map to neighbor
  route-reflector-client   Configure a neighbor as Route Reflector client
  send-community           Send community attribute to this neighbor
  unsuppress-map           Route-map to selectively unsuppress suppressed
                           routes
  weight                   Set default weight for routes from this neighbor
```

Commands entered in BGP global configuration mode apply to the IPv4 unicast address family. Where relevant, this chapter discusses and provides IPv4-unicast-specific examples. You must first configure IPv4 unicast routing for any IPv4 routing protocol to be active.

The following neighbor configuration options are allowed under BGP global configuration mode:

```
device(config-bgp)# neighbor 10.1.1.1 ?
Possible completions:
```

```
 activate                 Allow exchange of route in the current family mode
additional-paths         Specify bgp additional paths
advertisement-interval   Minimum interval between sending BGP routing updates
allowas-in               Accept as-path with my AS present in it
as-override              Override matching AS-number while sending update
bfd                      Set BFD parameters for BGP4 neighbor
capability               Advertise capability to the peer
default-originate        Originate default route to peer
description              Neighbor by description
distribute-list          Apply Address Filters to neighbor
ebgp-btsh                Enable EBGP TTL Security Hack Protection
ebgp-multihop            Allow EBGP neighbors not on directly connected
                         networks
enforce-first-as         Enforce the first AS for EBGP routes
fail-over                Set Failover source for BGP4 for this peer
filter-list              Establish BGP filters
local-as                 Assign local-as number to neighbor
maxas-limit              Impose limit on number of ASes in AS-PATH attribute
maximum-prefix           Maximum number of prefix accept from this peer
next-hop-self            Disable the next hop calculation for this neighbor
password                 Enable TCP-MD5 password protection
peer-group               Assign peer-group to neighbor
prefix-list              Prefix List for filtering routes
remote-as                Specify a BGP neighbor
remove-private-as        Remove private AS number from outbound updates
route-map                Apply route map to neighbor
route-reflector-client   Configure a neighbor as Route Reflector client
send-community           Send community attribute to this neighbor
shutdown                 Administratively shut down this neighbor
soft-reconfiguration     Per neighbor soft reconfiguration
static-network-edge      Neighbor as special service edge, static-network
                         shall not be advertised if installed as DROP
timers                   BGP per neighbor timers
unsuppress-map           Route-map to selectively unsuppress suppressed routes
update-source            Source of routing updates
weight                   Set default weight for routes from this neighbor
```

Configuring BGP4 neighbors

BGP4 neighbors can be configured using this procedure.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   device(config)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   device(config-bgp)# local-as 1000
   ```

5. Enter the **neighbor remote-as** command, and specify an IP address, to specify the ASN in which the remote neighbor resides.

   ```
   device(config-bgp-router)# neighbor 10.1.1.1 remote-as 1001
   device(config-bgp)# neighbor 10.1.1.1 remote-as 1001
   ```

The following example configures a BGP4 neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 1001

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 1001

device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# neighbor 10.1.1.1 remote-as 1001
```

Requiring the first AS to be the neighbor AS

By default, the Extreme device does not require the first AS listed in the AS_SEQUENCE field of an AS path update message from eBGP neighbors be the AS of the neighbor that sent the update.

You can configure the Extreme device to require that the first AS listed in the AS_SEQUENCE field of an AS path update message from eBGP neighbors is the AS of the neighbor that sent the update. You can enable this feature globally on the device, or for a specific neighbor or peer group. The device accepts the update only if the AS numbers match. If the AS numbers do not match, the Extreme device sends a notification message to the neighbor and closes the session. This requirement applies to all updates received from eBGP neighbors.

The hierarchy for enforcement of this feature is as follows:

- If the enforce-first-as value is configured, a neighbor attempts to use this value.
- If none is configured, the neighbor attempts to use the configured value for a peer group.
- If neither configuration exists, enforcement is the global configuration which is disabled by default.

The following example enables this feature globally.

```
device(config-bgp)# enforce-first-as
```

The following example enables this feature for a specific neighbor.

```
device(config-bgp)# neighbor 10.1.1.1 enforce-first-as enable
```

The following example enables this feature for a peer group.

```
device(config-bgp)# neighbor Peergroup1 enforce-first-as enable
```

The following example enables this feature globally, for a peer group, and a neighbor.

```
device(config)# router bgp
BGP4: Please configure 'local-as' parameter in order to enable BGP4.
device(config-bgp)# local-as 1

device(config-bgp)# enforce-first-as
device(config-bgp)# neighbor abc peer-group
device(config-bgp)# neighbor abc remote-as 2
device(config-bgp)# neighbor abc enforce-first-as disable
device(config-bgp)# neighbor 192.168.1.2 peer-group abc
device(config-bgp)# neighbor 192.168.1.2 enforce-first-as enable
```

For a neighbor or peer configuration, when the first-as requirement is enabled, its status appears in the output of the **show running configuration** command. The optional **enable** or **disable** lets you specify whether the output of the **show running configuration** command includes the configuration of the first-as requirement to show what you have actually configured.

Peer groups

Neighbors having the same attributes and parameters can be grouped together by means of the **neighbor peer-group** command. You must first create a peer-group, after which you can associate neighbor IP addresses with the peer-group. All of the attributes that are allowed on a neighbor are also allowed on a peer-group.

The benefits of peer groups are:

- Simplified neighbor configuration - You can configure a set of neighbor parameters and then apply them to multiple neighbors. You do not need to configure the common parameters individually on each neighbor.
- Flash memory conservation - Using peer groups instead of individually configuring all the parameters for each neighbor requires fewer configuration commands in the startup configuration file.

You can perform the following tasks on a peer-group basis:

- Reset neighbor sessions
- Perform soft-outbound resets (the device updates outgoing route information to neighbors but does not entirely reset the sessions with those neighbors)
- Clear BGP4 message statistics
- Clear error buffers

An attribute value configured explicitly for a neighbor takes precedence over the attribute value configured for a peer-group. If neither the peer-group nor the individual neighbor has the attribute configured, the default value for the attribute is used.

For the parameters of a peer group to take effect, the peer group must be activated in the IPv4 or IPv6 address-family. By default, only IPv4 unicast address family is activated for a peer-group. A user needs to explicitly activate a peer-group in the IPv6 unicast address-family configuration mode when used with IPv6 peers.

Configuring BGP4 peer groups

A peer group can be created and neighbor IPv4 addresses can be associated with the peer group.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   device(config)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   device(config-bgp)# local-as 1000
   ```

5. Enter the **neighbor** *peer-group-name* **peer-group** command to create a peer group.

   ```
   device(config-bgp-router)# neighbor mypeergroup1 peer-group
   device(config-bgp)# neighbor mypeergroup1 peer-group
   ```

6. Enter the **neighbor** *peer-group-name* **remote-as** command to specify the ASN of the peer group.

```
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
```

```
device(config-bgp)# neighbor mypeergroup1 remote-as 11
```

7. Enter the **neighbor** *ip-address* **peer-group** command to associate a neighbor with the peer group.

```
device(config-bgp-router)# neighbor 10.2.2.2 peer-group mypeergroup1
```

```
device(config-bgp)# neighbor 10.2.2.2 peer-group mypeergroup1
```

8. Enter the **neighbor** *ip-address* **peer-group** command to associate another neighbor with the peer group.

```
device(config-bgp-router)# neighbor 10.3.3.3 peer-group mypeergroup1
```

```
device(config-bgp)# neighbor 10.3.3.3 peer-group mypeergroup1
```

The following example creates a peer group and specifies two neighbors to belong to the peer group.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor mypeergroup1 peer-group
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
device(config-bgp-router)# neighbor 10.2.2.2 peer-group mypeergroup1
device(config-bgp-router)# neighbor 10.3.3.3 peer-group mypeergroup1
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor mypeergroup1 peer-group
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
device(config-bgp-router)# neighbor 10.2.2.2 peer-group mypeergroup1
device(config-bgp-router)# neighbor 10.3.3.3 peer-group mypeergroup1
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# neighbor mypeergroup1 peer-group
device(config-bgp)# neighbor mypeergroup1 remote-as 11
device(config-bgp)# neighbor 10.2.2.2 peer-group mypeergroup1
device(config-bgp)# neighbor 10.3.3.3 peer-group mypeergroup1
```

Advertising the default BGP4 route

By default, a BGP device does not originate and advertise a default route using BGP4. A BGP4 default route is the IP address 0.0.0.0 and the route prefix 0 or network mask 0.0.0.0. For example, 0.0.0.0/0 is a default route. A BGP device can be configured to advertise the default IPv4 route to all BGP4 neighbors and to install that route in the local BGP4 route table.

The default route must be present in the local IPv4 route table.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **address-family ipv4 unicast** command to enter IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **default-information-originate** command to advertise the default IPv4 route to all BGP4 neighbors and to install that route in the local BGP4 route table.

```
device(config-bgp-ipv4u)# default-information-originate
```

```
device(config-bgp)# default-information-originate
```

The following example enables a BGP4 device to advertise the default IPv4 route to all BGP4 neighbors and to install that route in the local BGP4 route table.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# default-information-originate

device# configure terminal
device(config)# router bgp
device(config-bgp)# default-information-originate

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# default-information-originate
```

Four-byte AS numbers

Four-byte autonomous system numbers (ASNs) can be optionally configured on a device, peer-group, or neighbor. If this is enabled, the device announces and negotiates "AS4" capability with its neighbors.

You can configure AS4 capability to be enabled or disabled either at the BGP global level or at the neighbor or peer-group level.

You can configure AS4 capability to be enabled for a neighbor while still keeping AS4 numbers disabled at the global level, or vice-versa. The neighbor AS4 capability configuration takes precedence. If AS4 capability is not configured on the neighbor, then the peer-group configuration takes effect. The global configuration is used if AS4 capability is configured neither at the neighbor nor at the peer-group level. If a device having a 4-byte ASN tries to connect to a device that does not have AS4 support, peering will not be established.

Enabling ASN capability globally

Four-byte autonomous system numbers (ASNs) can be configured on a device. The following task enables 4-byte autonomous system number (ASN) capability at the BGP global level.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **capability as4-enable** command to enable 4-byte autonomous system number (ASN) capability globally.

```
device(config-bgp-router)# capability as4-enable
```

The following example enables 4-byte autonomous system number (ASN) capability globally.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# capability as4-enable
```

Enabling ASN capability for a BGP4 neighbor

Four-byte autonomous system numbers (ASNs) can be configured for a neighbor or peer-group. The following task enables 4-byte autonomous system number (ASN) capability for a neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

4. Enter the **neighbor capability as4** command with the **enable** parameter, and specify an IP address, to enable 4-byte autonomous system number (ASN) capability for a specific neighbor.

```
device(config-bgp-router)# neighbor 10.1.2.3 capability as4 enable
```

The following example enables 4-byte autonomous system number (ASN) capability for a specific neighbor.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.2.3 capability as4 enable
```

Cooperative BGP4 route filtering

By default, the device performs all filtering of incoming routes locally, on the device itself. You can use cooperative BGP4 route filtering to cause the filtering to be performed by a neighbor before it sends the routes to the device. Cooperative filtering conserves resources by eliminating unnecessary route updates and filter processing. For example, the device can send a deny filter to a neighbor, which the neighbor uses to filter out updates before sending them to the device. The neighbor saves the resources it would otherwise use to generate the route updates, and the device saves the resources it would use to filter out the routes.

When you enable cooperative filtering, the device advertises this capability in its Open message to the neighbor when initiating the neighbor session. The Open message also indicates whether the device is configured to send filters, receive filters, or both, and the types of filters it can send or receive. The device sends the filters as Outbound Route Filters (ORFs) in route refresh messages.

To configure cooperative filtering, perform the following tasks on the device and on the BGP4 neighbor:

- Configure the filter.

NOTE
Cooperative filtering is currently supported only for filters configured using IP prefix lists.

- Apply the filter as an inbound filter to the neighbor.
- Enable the cooperative route filtering feature on the device. You can enable the device to send ORFs to the neighbor, to receive ORFs from the neighbor, or both. The neighbor uses the ORFs you send as outbound filters when it sends routes to the device. Likewise, the device uses the ORFs it receives from the neighbor as outbound filters when sending routes to the neighbor.
- Reset the BGP4 neighbor session to send and receive ORFs.
- Perform these steps on the other device.

NOTE
If the device has inbound filters, the filters are still processed even if equivalent filters have been sent as ORFs to the neighbor.

BGP4 parameters

Some parameter changes take effect immediately while others do not take full effect until the device sessions with its neighbors are reset. Some parameters do not take effect until the device is rebooted.

The following parameter changes take effect immediately:

- Enable or disable BGP4.
- Set or change the local AS.
- Add neighbors.
- Change the update timer for route changes.
- Disable or enable fast external failover.
- Specify individual networks that can be advertised.
- Change the default local preference, default information originate setting, or administrative distance.
- Enable or disable use of a default route to resolve a BGP4 next-hop route.
- Enable or disable MED (metric) comparison.
- Require the first AS in an update from an EBGP neighbor to be the neighbor AS.
- Change MED comparison parameters.
- Disable comparison of the AS-Path length.
- Enable comparison of the device ID.
- Enable next-hop recursion.
- Change the default metric.
- Disable or re-enable route reflection.
- Configure confederation parameters.
- Disable or re-enable load sharing.
- Change the maximum number of load sharing paths.
- Change other load-sharing parameters.
- Define route flap dampening parameters.
- Add, change, or negate redistribution parameters (except changing the default MED).

- Add, change, or negate route maps (when used by the **network** command or a redistribution command).
- Apply maximum AS path limit settings for UPDATE messages.
- Aggregate routes
- Add, change, or negate filter tables that affect inbound and outbound route policies.

The following parameter changes take effect only after the BGP4 sessions on the device are cleared, or reset using the "soft" clear option:

- Change the Hold Time or Keep Alive Time.
- Aggregate routes
- Add, change, or negate filter tables that affect inbound and outbound route policies.
- Apply maximum AS path limit settings to the RIB.

The following parameter change takes effect only after you disable and then re-enable redistribution:

- Change the default MED (metric).

Route redistribution

The redistribution of static, connected, RIP, ISIS, and OSPF routes into BGP is supported. Similarly, routes learned through BGP can also be redistributed into OSPF.

The redistribution of static, connected, ISIS, and OSPF routes into BGP is supported. Similarly, routes learned through BGP can also be redistributed into OSPF or IS-IS.

The redistribution of static, connected, OSPF routes into BGP is supported. Similarly, routes learned through BGP can also be redistributed into OSPF.

The redistribution of static, connected, and OSPF routes into BGP is supported. Similarly, routes learned through BGP can also be redistributed into OSPF.

The redistribution of static, connected, and OSPF routes into BGP is supported. Similarly, routes learned through BGP can also be redistributed into OSPF.

An optional route-map can be specified, and this map will be consulted before routes are added to BGP. Management routes are not redistributed.

Redistributing routes into BGP4

Various routes can be redistributed into BGP. This tasks redistributes connected routes into BGP4.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   device(config)# router bgp
   ```

4. Enter the **address-family ipv4 unicast** command to enter BGP IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **redistribute** command using the **connected** keyword to redistribute connected routes.

```
device(config-bgp-ipv4u)# redistribute connected
```

```
device(config-bgp)# redistribute connected
```

The following example redistributes connected routes into BGP4.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# redistribute connected

device# configure terminal
device(config)# router bgp
device(config-bgp)# redistribute connected

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# redistribute connected
```

Advertised networks

As previously described, you can advertise routes into BGP by redistributing static, connected, RIP, ISIS, or OSPF routes.

As previously described, you can advertise routes into BGP by redistributing static, connected, ISIS, or OSPF routes.

As previously described, you can advertise routes into BGP by redistributing static, connected, or OSPF routes.

As previously described, you can advertise routes into BGP by redistributing static, connected, or OSPF routes.

As previously described, you can advertise routes into BGP by redistributing static, connected, or OSPF routes.

However, you can explicitly specify routes to be advertised by BGP4 by using the **network** command in IPv4 address-family unicast configuration mode.

However, you can explicitly specify routes to be advertised by BGP4 by using the **network** command in BGP global configuration mode or IPv4 address-family multicast configuration mode.

With the exception of static network routes, the routing table must have this route already installed before BGP4 can advertise this route. You can also specify a route to be local. If the same route is received by means of eBGP, the local IGP route will be preferred. You can also specify a weight that the device adds to routes that are received from the specified BGP neighbor. BGP4 prefers larger weights over smaller weights.

Refer to the *Extreme NetIron Command Reference* for configuration examples and more information.

Refer to the *Extreme Network OS Command Reference*for configuration examples and more information.

Refer to the *Extreme SLX-OS Command Reference* for more information.

Importing routes into BGP4

Routes can be explicitly specified for advertisement by BGP.

With the exception of static network routes, the routes imported into BGP4 must first exist in the IPv4 unicast route table.

With the exception of static network routes, the routes imported into BGP4 must first exist in the IPv4 multicast or unicast route table.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

   ```
   device(config)# router bgp
   ```

4. Enter the **neighbor remote-as** command to specify the ASN in which the remote neighbor resides.

   ```
   device(config-bgp-router)# neighbor 10.2.2.2 remote-as 1001
   ```

   ```
   device(config-bgp)# neighbor 10.2.2.2 remote-as 1001
   ```

5. Enter the **address-family ipv4 unicast** command to enter address family IPv4 unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

6. Enter the **network** command and specify a *network/mask* to import the specified prefix into the BGP4 database.

   ```
   device(config-bgp)# network 10.1.1.1/32
   ```

   ```
   device(config-bgp-ipv4u)# network 10.1.1.1/32
   ```

The following example imports the 10.1.1.1/32 prefix in to the BGP4 database for advertising.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# neighbor 10.2.2.2 remote-as 1001
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# network 10.1.1.1/32

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# neighbor 10.2.2.2 remote-as 1001
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# network 10.1.1.1/32

device# configure terminal
device(config)# router bgp
device(config-bgp)# 10.2.2.2 remote-as 1001
device(config-bgp)# network network 10.1.1.1/32
```

Static networks

Before advertising any route, BGP checks for its existence in the routing table. BGP can be configured to advertise a stable route that does not depend on its existence in the routing table.

This allows you to configure a static network in BGP4, creating a stable BGP4 network in the core. While a route configured with this feature will never flap unless it is manually deleted, a "static" BGP4 network will not interrupt the normal BGP4 decision process on other learned routes being installed into the Routing Table Manager (RTM). Consequently, when there is a route that can be resolved, it will be installed into the RTM.

When the configured route is lost, BGP installs the "null0" route in the routing table. Later, when the route is resolved, the null0 route is removed.

> NOTE
> The BGP4 network route and the BGP4 static network route are mutually exclusive. They cannot be configured with the same prefix and mask.

Configuring a static network

BGP can be configured to advertise a stable route that does not depend on its existence in the routing table. The following task configures a static network and sets the administrative distance.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
device(config)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
device(config-bgp)# local-as 1000
```

5. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

6. Enter the **static-network** command with the **distance** parameter, and specify a value, to configure a static network and set an administrative distance.

```
device(config-bgp-ipv4u)# static-network 10.11.12.0/32 distance 300
device(config-bgp-ipv4u)# static-network 10.11.12.0/32 distance 200
device(config-bgp)# static-network 10.11.12.0/32 distance 300
```

The following example configures a static network and sets an administrative distance of 300.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# static-network 10.11.12.0/32 distance 300
```

The following example configures a static network and sets an administrative distance of 300.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# static-network 10.11.12.0/32 distance 300
```

The following example configures a static network and sets an administrative distance of 300.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# static-network 10.11.12.0/32 distance 300
```

The following example configures a static network and sets an administrative distance of 200.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# static-network 10.11.12.0/32 distance 200
```

Route reflection

A BGP device can act as a route-reflector client or as a route reflector. You can configure a BGP peer as a route-reflector client from the device that is going to reflect the routes and act as the route reflector using the **neighbor route-reflector-client** command.

When there is more than one route reflector, they should all belong to the same cluster. By default, the value for **cluster-id** is used as the device ID. The device ID can be changed using the **cluster-id** command.

The route-reflector server reflects the routes as follows:

- Routes from the client are reflected to the client as well as to nonclient peers.
- Routes from nonclient peers are reflected only to client peers.

If route-reflector clients are connected in a full iBGP mesh, you can disable client-to-client reflection on the route reflector using the **no client-to-client-reflection** command.

A BGP device advertises only those routes that are preferred ones and are installed into the Routing Table Manager (RTM). When a route cannot be installed into the RTM because the routing table is full, the route reflector may not reflect that route. In cases where the route reflector is not placed directly in the forwarding path, you can configure the route reflector to reflect routes even though those routes are not in the RTM using the **always-propagate** command.

Configuring a cluster ID for a route reflector

The cluster ID can be changed if there is more than one route reflector, so that all route reflectors belong to the same cluster.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

```
device(config-bgp)# local-as 1000
```

5. Enter the **cluster-id** command and specify a value to change the cluster ID of a device from the default device ID.

```
device(config-bgp-router)# cluster-id 321
```

```
device(config-bgp)# cluster-id 321
```

The following example changes the cluster ID of a device from the default device ID to 321.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# cluster-id 321

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# cluster-id 321

device# configure terminal
device(config)# router bgp
device(config-bgp)# cluster-id 321
```

Configuring a route reflector client

A BGP peer can be configured as a route reflector client.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

```
device(config-bgp)# local-as 1000
```

5. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

6. Enter the **neighbor route-reflector-client** command to configure a specified neighbor to be a route reflector client.

```
device(config-bgp-ipv4u)# neighbor 10.1.1.1 route-reflector-client
```

```
device(config-bgp)# neighbor 10.1.1.1 route-reflector-client
```

The following example configures a neighbor with the IPv4 address 10.1.1.1 to be a route reflector client.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# neighbor 10.1.1.1 route-reflector-client

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.1.1 route-reflector-client

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.1.1 route-reflector-client
```

Route flap dampening

A route flap is a change in the state of a route, from up to down or down to up. A route state change causes changes in the route tables of the devices that support the route.

Frequent route state changes can cause Internet instability and add processing overhead to the devices that support the route. Route flap dampening helps reduce the impact of route flap by changing the way a BGP4 device responds to route state changes. When route flap dampening is configured, the device suppresses unstable routes until the number of route state changes drops enough to meet an acceptable degree of stability.

Route flap dampening is disabled by default. You can enable the feature globally or on an individual route basis using route maps.

> NOTE
> The device applies route flap dampening only to routes learned from eBGP neighbors.

The route flap dampening mechanism is based on penalties. When a route exceeds a configured penalty value, the device stops using that route and stops advertising it to other devices. The mechanism also allows route penalties to reduce over time if route stability improves.

Aggregating routes advertised to BGP neighbors

A device can be configured to aggregate routes in a range of networks into a single IP prefix.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **address-family ipv4 unicast** command to enter BGP address family IPv4 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **aggregate-address** command to aggregate the routes from a range of networks into a single network prefix.

```
device(config-bgp-ipv4u)# aggregate-address 10.1.1.1/32
```

```
device(config-bgp)# aggregate-address 10.1.1.1/32
```

The following example enables a BGP4 device to advertise the default route and send the default route to a specified neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# aggregate-address 10.1.1.1/32
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# aggregate-address 10.1.1.1/32
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# aggregate-address 10.1.1.1/32
```

Advertising the default BGP4 route

By default, a BGP device does not originate and advertise a default route using BGP4. A BGP4 default route is the IP address 0.0.0.0 and the route prefix 0 or network mask 0.0.0.0. For example, 0.0.0.0/0 is a default route. A BGP device can be configured to advertise the default IPv4 route to all BGP4 neighbors and to install that route in the local BGP4 route table.

The default route must be present in the local IPv4 route table.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **address-family ipv4 unicast** command to enter IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **default-information-originate** command to advertise the default IPv4 route to all BGP4 neighbors and to install that route in the local BGP4 route table.

```
device(config-bgp-ipv4u)# default-information-originate
```

```
device(config-bgp)# default-information-originate
```

The following example enables a BGP4 device to advertise the default IPv4 route to all BGP4 neighbors and to install that route in the local BGP4 route table.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# default-information-originate
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# default-information-originate
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# default-information-originate
```

Advertising the default BGP4 route to a specific neighbor

A BGP device can be configured to advertise the default IPv4 route to a specific neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

```
device(config-bgp)# local-as 1000
```

5. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

6. Enter the **neighbor default-originate** command and specify an IP address to enable the BGP4 device to advertise the default IPv4 route to a specific neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.4.4.4 default-originate
```

```
device(config-bgp)# neighbor 10.4.4.4 default-originate
```

The following example enables a BGP4 device to advertise the default IPv4 route to a specific neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.4.4.4 default-originate

device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# neighbor 10.4.4.4 default-originate

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.4.4.4 default-originate
```

Multipath load sharing

Unlike IGP, BGP does not perform multipath load sharing by default. Therefore, the maximum number of paths across which BGP can balance the traffic is set to 1 by default. You can change this value by using the **maximum-paths** command.

By default, when BGP4 multipath load sharing is enabled, both iBGP and eBGP paths are eligible for load sharing, while paths from different neighboring autonomous systems are not eligible. You can change load sharing to apply only to iBGP or eBGP paths, or to support load sharing among paths from different neighboring autonomous systems.

If the maximum-path value must be picked up from the **ip load-sharing** configuration on the router, use the following:

```
sw0(config-bgp-ipv4u)# maximum-paths use-load-sharing
```

NOTE

The **ip load-sharing** command has been deprecated. Note the following use cases with respect to legacy configurations:

1. If the **ip load-sharing** command has been used in a previous release and maximum paths **has not** been configured, then load balancing is influenced only for static routes and load balancing for the routing protocol is not affected.
2. If the **ip load-sharing** command has been used in a previous release and maximum paths **has** been configured, then the lowest configured number of paths takes precedence.
3. With the current release, if only the **ip load-sharing** command has been used with an existing configuration and maximum paths **has not** been configured, then there is no effect on load sharing for the routing protocol.
4. With the current release, if only the **ip load-sharing** command has been used with an existing configuration and maximum paths **has** been configured, then the lowest configured number of paths takes precedence. The number of maximum paths can still be adjusted, by means of the **maximum-paths** command, prior to an upgrade to the current release.

Specifying the weight added to received routes

The weight that the device adds to received routes can be specified. The following task changes the weight from the default for routes that are received from a specified BGP neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

```
device(config-bgp)# local-as 65520
```

5. Enter the **address-family ipv4 unicast** command to enter address family IPv4 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

6. Enter the **neighbor weight** command and specify an *ip address* and a weight value to specify a weight that the device adds to routes that are received from the specified BGP4 neighbor.

```
device(config-bgp)# neighbor 10.11.12.13 weight 100
```

```
device(config-bgp-ipv4u)# neighbor 10.11.12.13 weight 100
```

The following example specifies a weight of 100 that the device adds to routes that are received from the specified BGP4 neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.11.12.13 weight 100
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.11.12.13 weight 100
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# neighbor 10.11.12.13 weight 100
```

Using the IPv4 default route as a valid next hop for a BGP4 route

In certain cases, such as when a device is acting as an edge device, it can be configured to use the default route as a valid next hop.

By default, a device does not use a default route to resolve a BGP4 next-hop route. If the IPv4 route lookup for the BGP4 next-hop does not result in a valid IGP route (including static or direct routes), the BGP4 next-hop is considered to be unreachable and the BGP4 route is not used. You can configure the device to use the default route as a valid next hop.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **address-family ipv4 unicast** command to enter IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **next-hop-enable-default** command to configure the device to use the default route as a valid next hop.

```
device(config-bgp)# next-hop-enable-default
```

```
device(config-bgp-ipv4u)# next-hop-enable-default
```

The following example configures a BGP4 device to use the default route as a valid next hop.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# next-hop-enable-default
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# next-hop-enable-default
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# next-hop-enable-default
```

Adjusting defaults to improve routing performance

The following examples illustrate a variety of options for enabling and fine-tuning route flap dampening.

The following example enables default dampening as an address-family function.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# dampening
```

```
device# configure terminal
device(config)# rbridge-id 10
device(config-rbridge-id-10)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# dampening
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# dampening
```

The following example changes all dampening values.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# dampening 20 200 2500 40

device# configure terminal
device(config)# rbridge-id 10
device(config-rbridge-id-10)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# dampening 20 200 2500 40

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# dampening 20 200 2500 40
```

Next-hop recursion

For each BGP4 route learned, the device performs a route lookup to obtain the IPv4 address of the next hop for the route. A BGP4 route is eligible for addition in the IPv4 route table only if the following conditions are true:

- The lookup succeeds in obtaining a valid next-hop IPv4 address for the route.
- The path to the next-hop IP address is an IGP path or a static route path.

By default, only one lookup is performed for the next-hop IPv4 address for the BGP4 route. If the next hop lookup does not result in a valid next hop IPv4 address, or the path to the next hop IPv4 address is a BGP4 path, the BGP4 route destination is considered unreachable. The route is not eligible to be added to the IPv4 route table.

The BGP4 route table can contain a route with a next hop IPv4 address that is not reachable through an IGP route, even though the device can reach a hop farther away through an IGP route. This can occur when the IGPs do not learn a complete set of IGP routes, so the device learns about an internal route through iBGP instead of through an IGP. In this case, the IPv4 route table does not contain a route that can be used to reach the BGP4 route destination.

When next-hop recursion is enabled, if the lookup for the next hop IP address results in an iBGP path that originated in the same AS, then the next hop is considered as resolved and BGP4 depended routes are eligible for addition in the IPV4 route table.

Enabling next-hop recursion

Next hop recursion can be enabled so that a device can find the IGP route to the next hop gateway for a BGP4 route.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   device(config)# router bgp
   ```

4. Enter the **address-family ipv4 unicast** commandto enter IPv4 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

5. Enter the **next-hop-recursion** command to enable recursive next hop lookups.

```
device(config-bgp)# next-hop-recursion

device(config-bgp-ipv4u)# next-hop-recursion
```

The following example enables recursive next hop lookups.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# next-hop-recursion

device# configure terminal
device(config)# router bgp
device(config-bgp)# next-hop-recursion

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# next-hop-recursion
```

Enabling next-hop recursion in the IPv4 multicast address family

Next hop recursion can be enabled so that a device can find the IGP route to the next hop gateway for a BGP4 route.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv4 multicast** command to enter address family IPv4 multicast configuration mode.

```
device(config-bgp)# address-family ipv4 multicast
```

4. Enter the **next-hop-recursion** command to enable recursive next hop lookups.

```
device(config-bgp-ipv4m)# next-hop-recursion
```

The following example enables recursive next hop lookups.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 multicast
device(config-bgp-ipv4m)# next-hop-recursion
```

Route filtering

The following route filters are supported:

- AS-path filter
- Community filter
- Prefix list
- Route map
- Table map

NOTE
Support for access lists in route filtering is not available, and has been replaced by prefix-list filtering. BGP does not use community and extended-community filters directly. Rather, it uses them indirectly through route-map filtering by means of the **route-map** command.

NOTE
BGP does not use community and extended-community filters directly. Rather, it uses them indirectly through route-map filtering by means of the **route-map** command.

BGP VRF route filters

Filtering based on route-map policies can be added to the EVPN IPv4 and IPv6 prefix routes that are imported to or exported from VRFs.

The IPv4/IPv6 prefix routes of a given VRF instance in the BGP routing table can be imported from or exported to the EVPN routing table by means of the **route target (EVPN)** command.

Import routes

The **route-target (VRF)** command is used to import the corresponding *x:y* route-target extended-community-matched IPv4 or IPv6 prefix routes that are learned from the EVPN table to the VRF BGP table. The **import map evpn** command is then used to apply the filtering policies to all those routes before they are imported. Filtering is also supported for EVPN host routes (ARP/ND routes imported from EVPN to BGP VRF as /32 or /128 prefix routes). When the VRF import map is configured or removed, or additions, modifications, or deletions are made to the corresponding route-map, the **clear bgp evpn neighbor all soft in** command is effectively applied automatically to all EVPN neighbors.

Export routes

The **route-target (VRF)** command is also used to export IPv4 or IPv6 prefix routes from the VRF BGP table to the EVPN table with a route-target extended-community value of *x:y*. The **export map evpn** command is then used to apply the filtering policies to all the exported routes. When the VRF export map is configured or removed, or additions, modifications, or deletions are made to the corresponding route-map, the EVPN table is refreshed for the appropriate address family, IPv4 or IPv6.

Regular-expression-based extended-community match filters

The **ip extcommunity-list** command now supports regular-expression matching of route target or site of origin values to filter VRF routes. The previous implementation supported the **rt** (route target) and **soo** (site of origin) keywords, based on standard extended-community match filters where the **rt** and **soo** values are matched exactly. The range of standard extended-community match filters is from 1 through 99. The range of expanded extended-community match filters is from 100 through 500.

A regular expression is entered as part of a command and is a pattern made up of symbols, letters, and numbers that represent an input string for matching (or sometimes not matching). Matching the string to the specified pattern is called *pattern matching*.

Pattern matching either succeeds or fails. If a regular expression can match two different parts of an input string, it will match the earliest part first.

NOTE
For details, refer to "BGP regular expression pattern-matching characters" in this document.

Example standard and expanded extended-community configuration

The following illustrates an extended-community list configuration that includes both standard and expanded match filters.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# ip extcommunity-list 1 permit rt 1:1 rt 1:2 rt 1:3 rt 1:4
device(config-rbridge-id-1)# ip extcommunity-list 2 permit rt 1:9 soo 1:7
device(config-rbridge-id-1)# ip extcommunity-list 100 permit 6513:*
device(config-rbridge-id-1)# ip extcommunity-list 101 permit 1:[2-6]
device(config-rbridge-id-1)# route-map map1 permit 1
device(config-routemap-map1/permit/1)# match extcommunity 1 2 100 101
```

The incoming route has to match either one of the four extended community lists: 1 or 2 or 100 or 101 (logical OR).

The incoming route has to match all the values within any one of the four extended community lists configured above (logical AND).

Using BGP VRF route filters

This task illustrates the configuration of BGP VRF import and export route filters in conjunction with a standard and an expanded extended-community list.

1. Enter global configuration mode.

   ```
   device# configure terminal
   device(config)#
   ```

2. Enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 1
   device(config-rbridge-id-1)#
   ```

3. Specify a VRF instance and enter VRF configuration mode.

   ```
   device(config-rbridge-id-1)# vrf red
   device(config-vrf-red)#
   ```

4. Enter VRF address-family IPv4 unicast configuration mode.

   ```
   device(config-vrf-red)# address-family ipv4 unicast
   device(vrf-red-ipv4-unicast)#
   ```

   NOTE
   Use the **address-family ipv6 unicast** command for IPv6 configurations.

5. Use the **import map evpn** command to apply the route map "myimportmap" to imported prefixes.

   ```
   device(vrf-red-ipv4-unicast)# import map myimportmap evpn
   ```

6. Use the **export map evpn** command to apply the route map "myexportmap" to exported prefixes.

   ```
   device(vrf-red-ipv4-unicast)# export map myexportmap evpn
   ```

7. Return to RBridge ID configuration mode and configure a standard extended-community list (range is from 1 through 99), using the **rt** (route target) and **soo** (site of origin) keywords.

   ```
   device(config-rbridge-id-1)# ip extcommunity-list 1 permit rt 1:9 soo 1:7
   ```

8. Optionally, configure an expanded extended-community list by specifying a regular-expression match pattern.

```
device(config-rbridge-id-1)# ip extcommunity-list 100 permit 6513:*
```

NOTE
For details, refer to "BGP regular expression pattern-matching characters" in this document.

BGP regular expression pattern-matching characters

The following table illustrates the functions of BGP regular expression pattern-matching characters and illustrates their use.

NOTE
The **ip-extcommunity-list** command now supports a range of extended instances, from 100 through 500, beyond the standard range of 1 through 99.

TABLE 22 BGP regular expression pattern-matching characters

Regular expression character	Function	Examples
.	Matches any single character.	0.0 matches 0x0 and 020 t..t matches strings such as test, text, and tart
\	Matches the character following the backslash. Also matches (escapes) special characters.	172\.1\.. matches 172.1.10.10 but not 172.12.0.0 \. allows a period to be matched as a period
[ ]	Matches the characters or a range of characters separated by a hyphen, within left and right square brackets.	[02468a-z] matches 0, 4, and w, but not 1, 9, or K
^	Matches the character or null string at the beginning of an input string.	^123 matches 1234, but not 01234
?	Matches zero or one occurrence of the pattern. (Precede the question mark with Ctrl-V sequence to prevent it from being interpreted as a help command.)	ba?b matches bb and bab
$	Matches the character or null string at the end of an input string.	123$ matches 0123, but not 1234
*	Matches zero or more sequences of the character preceding the asterisk. Also acts as a wildcard for matching any number of characters.	5* matches any occurrence of the number 5 including none 18\..* matches the characters 18. and any characters that follow 18.
+	Matches one or more sequences of the character preceding the plus sign.	8+ requires there to be at least one number 8 in the string to be matched
() []	Nest characters for matching. Separate endpoints of a range with a dash (-).	(17)* matches any number of the two-character string 17 ([A-Za-z][0-9])+ matches one or more instances of letter-digit pairs: b8 and W4, as examples
|	Concatenates constructs. Matches one of the characters or character patterns on either side of the vertical bar.	A(B|C)D matches ABD and ACD, but not AD, ABCD, ABBD, or ACCD
_	Replaces a long regular expression list by matching a comma (,), left brace ({), right brace (}), the beginning of the input string, the end of the input string, or a space.	The characters _1300_ can match any of the following strings: ^1300$ ^1300space space1300 {1300, ,1300, {1300} ,1300,

Timers

The keep alive time specifies how frequently the device sends KEEPALIVE messages to its BGP4 neighbors. The hold time specifies how long the device waits for a KEEPALIVE or UPDATE message from a neighbor before concluding that the neighbor is dead. When the device concludes that a BGP4 neighbor is dead, the device ends the BGP4 session and closes the TCP connection to the neighbor.

A hold-time value of 0 means that the device waits indefinitely for messages from a neighbor without tearing down the session.

> NOTE
> Generally, you should set the hold time to three times the value of the keep alive time.

> NOTE
> You can override the global keep alive time and hold time on individual neighbors.

Enabling BGP4 in a non-default VRF

When BGP4 is enabled in a non-default VRF instance, the device enters BGP address-family IPv4 unicast VRF configuration mode. Several commands can then be accessed that allow the configuration of BGP4 for the non-default VRF instance.

A non-default VRF instance has been configured.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing globally.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

   ```
   device(config)# router bgp
   ```

4. Enter the **address-family ipv4 unicast** command with the **vrf** parameter, and specify a VRF name, to enter BGP address-family IPv4 unicast VRF configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast vrf red
   ```

   ```
   device(config-bgp)# address-family ipv4 unicast vrf red
   ```

The following example enables BGP address-family IPv4 unicast VRF configuration mode where several commands can be accessed that allow the configuration of BGP4 for the non-default VRF instance..

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast vrf red

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast vrf red
device(config-bgp-ipv4u-vrf)#

device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast vrf red
device(config-bgp-ipv4u-vrf)#
```

BGP4 outbound route filtering

The BGP4 Outbound Route Filtering Capability (ORF) feature is used to minimize the number of BGP updates sent between BGP peers.

When the ORF feature is enabled, unwanted routing updates are filtered out, reducing the amount of system resources required for generating and processing routing updates. The ORF feature is enabled through the advertisement of ORF capabilities to peer routers. The locally configured BGP4 inbound prefix filters are sent to the remote peer so that the remote peer applies the filter as an outbound filter for the neighbor.

The ORF feature can be configured with send and receive ORF capabilities. The local peer advertises the ORF capability in send mode, indicating that it will accept a prefix list from a neighbor and apply the prefix list to locally configured ORFs. The local peer exchanges the ORF capability in send mode with a remote peer for a prefix list that is configured as an inbound filter for that peer locally. The remote peer only sends the first update once it receives a ROUTEREFRESH request or BGP ORF with IMMEDIATE from the peer. The local and remote peers exchange updates to maintain the ORF on each router.

Configuring BGP4 outbound route filtering

The BGP4 Outbound Route Filtering (ORF) prefix list capability can be configured in receive mode, send mode, or both send and receive modes, minimizing the number of BGP updates exchanged between BGP peers.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   device(config)# router bgp
   ```

4. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

5. Enter the **neighbor prefix-list** command and specify the **in** keyword to filter the incoming route updates from a specified BGP neighbor.

   ```
   device(config-bgp)# neighbor 10.1.2.3 prefix-list myprefixlist in
   device(config-bgp-ipv4u)# neighbor 10.1.2.3 prefix-list myprefixlist in
   ```

6. Do one of the following:
   - Enter the **neighbor capability orf prefixlist** command and specify the **send** keyword to advertise ORF send capabilities.

     ```
     device(config-bgp)# neighbor 10.1.2.3 capability orf prefixlist send
     device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist send
     ```

   - Enter the **neighbor capability orf prefixlist** command and specify the **receive** keyword to advertise ORF receive capabilities.

     ```
     device(config-bgp)# neighbor 10.1.2.3 capability orf prefixlist receive
     device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist receive
     ```

   - Enter the **neighbor capability orf prefixlist** command to configure ORF capability in both send and receive modes.

     ```
     device(config-bgp)# neighbor 10.1.2.3 capability orf prefixlist
     device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist
     ```

The following example configures ORF in receive mode.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist receive

device# configure terminal
device(config)# router bgp
device(config-bgp)# neighbor 10.1.2.3 capability orf prefixlist receive

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist receive
```

The following example configures ORF in send mode.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# 10.1.2.3 prefix-list myprefixlist in
device(config-bgp-ipv4u)# 10.1.2.3 capability orf prefixlist send

device# configure terminal
device(config)# router bgp
device(config-bgp)# neighbor 10.1.2.3 prefix-list myprefixlist in
device(config-bgp)# neighbor 10.1.2.3 capability orf prefixlist send

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# 10.1.2.3 prefix-list myprefixlist in
device(config-bgp-ipv4u)# 10.1.2.3 capability orf prefixlist send
```

The following example configures ORF in both send and receive modes.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 prefix-list myprefixlist in
device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist

device# configure terminal
device(config)# router bgp
device(config-bgp)# neighbor 10.1.2.3 prefix-list myprefixlist in
device(config-bgp)# neighbor 10.1.2.3 capability orf prefixlist

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 prefix-list myprefixlist in
device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist
```

Enabling BGP4 cooperative route filtering

You can use cooperative BGP4 route filtering to cause the filtering to be performed by a neighbor before it sends the routes to the device, conserving resources by eliminating unnecessary route updates and filter processing. The following task enables cooperative route filtering.

NOTE
The current release supports cooperative filtering only for filters configured using IP prefix lists.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip prefix-list** command to to configure the IP prefix list instance.

```
device(config-rbridge-id-122)# ip prefix-list Routesfrom10234 deny 10.20.0.0/24
device(config-rbridge-id-122)# ip prefix-list Routesfrom10234 permit 10.0.0.0/0 le 32

device(config)# ip prefix-list Routesfrom10234 deny 10.20.0.0/24
device(config)# ip prefix-list Routesfrom10234 permit 10.0.0.0/0 le 32
```

4. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp

device(config)# router bgp
```

5. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

6. Enter the **neighbor prefix-list** command with the **in** parameter and specify a prefix-list to filter the incoming route updates from the specified BGP neighbor.

```
device(config-bgp)# neighbor 10.2.3.4 prefix-list Routesfrom1234 in

device(config-bgp-ipv4u)# neighbor 10.2.3.4 prefix-list Routesfrom1234 in
```

7. Enter the **capability orf prefixlist** command with the **send** parameter to enable the ORF prefix list capability in send mode.

```
device(config-bgp)# neighbor 10.2.3.4 capability orf prefixlist send
```

```
device(config-bgp-ipv4u)# neighbor 10.2.3.4 capability orf prefixlist send
```

The following example enables BGP4 cooperative route filtering.

```
device# configure terminal
device(config)# ip prefix-list Routesfrom10234 deny 10.20.0.0/24
device(config)# ip prefix-list Routesfrom10234 permit 10.0.0.0/0 le 32
device(config)# router bgp
device(config-bgp)# neighbor 10.2.3.4 prefix-list Routesfrom1234 in
device(config-bgp)# neighbor 10.2.3.4 capability orf prefixlist send
```

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip prefix-list Routesfrom10234 deny 10.20.0.0/24
device(config-rbridge-id-122)# ip prefix-list Routesfrom10234 permit 10.0.0.0/0 le 32
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.2.3.4 prefix-list Routesfrom1234 in
device(config-bgp-ipv4u)# neighbor 10.2.3.4 capability orf prefixlist send
```

```
device# configure terminal
device(config)# ip prefix-list Routesfrom10234 deny 10.20.0.0/24
device(config)# ip prefix-list Routesfrom10234 permit 10.0.0.0/0 le 32
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.2.3.4 prefix-list Routesfrom1234 in
device(config-bgp-ipv4u)# neighbor 10.2.3.4 capability orf prefixlist send
```

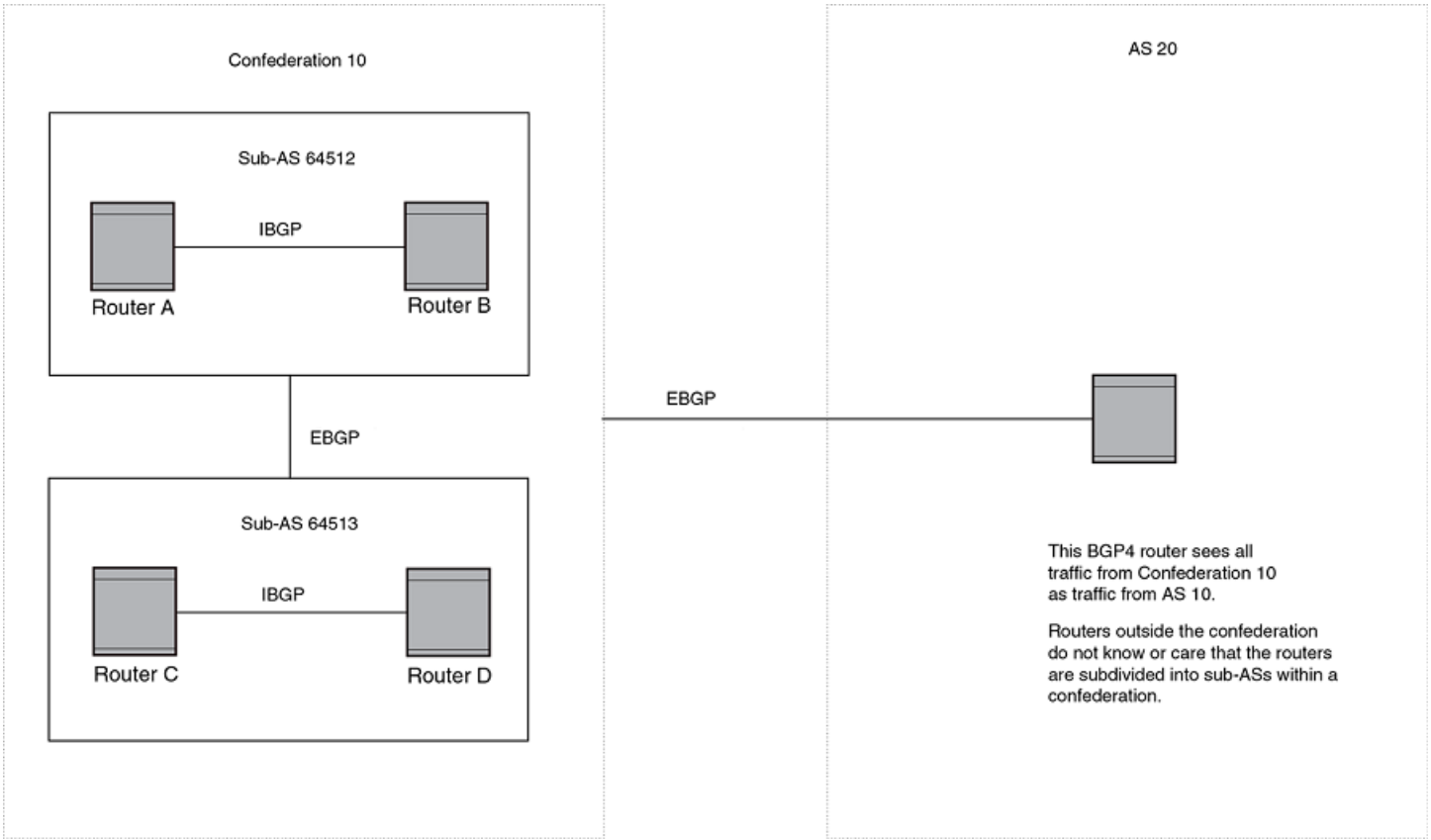
BGP4 confederations

A large autonomous system (AS) can be divided into multiple subautonomous systems and grouped into a single BGP4 confederation.

Each subautonomous system must be uniquely identified within the confederation AS by a subautonomous system number. Within each subautonomous system, all the rules of internal BGP (iBGP) apply. For example, all BGP routers inside the subautonomous system must be fully meshed. Although eBGP is used between subautonomous systems, the subautonomous systems within the confederation exchange routing information like iBGP peers. Next hop, Multi Exit Discriminator (MED), and local preference information is preserved when crossing subautonomous system boundaries. To the outside world, a confederation looks like a single AS.

The AS path list is a loop-avoidance mechanism used to detect routing updates leaving one subautonomous system and attempting to re-enter the same subautonomous system. A routing update attempting to re-enter a subautonomous system it originated from is detected because the subautonomous system sees its own subautonomous system number listed in the update's AS path.

FIGURE 36 Example BGP4 confederation



In this example, four devices are configured into two sub-autonomous systems, each containing two of the devices. The sub-autonomous systems are members of confederation 10. Devices within a sub-AS must be fully meshed and communicate using iBGP. In this example, devices A and B use iBGP to communicate. devices C and D also use IBGP. However, the sub-autonomous systems communicate with one another using eBGP. For example, device A communicates with device C using eBGP. The devices in the confederation communicate with other autonomous systems using eBGP.

Devices in other autonomous systems are unaware that devices A through D are configured in a confederation. In fact, when devices in confederation 10 send traffic to devices in other autonomous systems, the confederation ID is the same as the AS number for the devices in the confederation. Thus, devices in other autonomous systems see traffic as coming from AS 10 and are unaware that the devices in AS 10 are subdivided into sub-autonomous systems within a confederation.

Configuring BGP4 confederations

BGP4 confederations, composed of multiple subautonomous systems, can be created.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
device(config)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
device(config-bgp)# local-as 65520
```

5. Enter the **confederation identifier** command and specify an ASN to configure a BGP confederation identifier.

```
device(config-bgp-router)# confederation identifier 100
device(config-bgp)# confederation identifier 100
```

6. Enter the **confederation peers** command and specify as many ASNs as needed to list all BGP peers that will belong to the confederation.

```
device(config-bgp-router)# confederation peers 65520 65521 65522
device(config-bgp)# confederation peers 65520 65521 65522
```

The following example creates a confederation with the confederation ID “100” and adds three subautonomous systems to the confederation.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# confederation identifier 100
device(config-bgp-router)# confederation peers 65520 65521 65522

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# confederation identifier 100
device(config-bgp-router)# confederation peers 65520 65521 65522

device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# confederation identifier 100
device(config-bgp)# confederation peers 65520 65521 65522
```

BGP community and extended community

A BGP community is a group of destinations that share a common property. Community information identifying community members is included as a path attribute in BGP UPDATE messages. You can perform actions on a group using community and extended community attributes to trigger routing decisions.

All communities of a particular type can be filtered out, or certain values can be specified for a particular type of community. You can also specify whether a particular community is transitive or non-transitive across an autonomous system (AS) boundary.

An extended community is an 8-octet value and provides a larger range for grouping or categorizing communities. BGP extended community attributes are specified in RFC 4360.

You define the extended community list using the **ip extcommunity-list** command. The extended community can then be matched or applied to the neighbor through the route map. The route map must be applied on the neighbor to which routes need to carry the

extended community attributes. The "send-community" should be enabled for the neighbor configuration to start including the attributes while sending updates to the neighbor.

Defining BGP4 extended communities

In order to apply a BGP4 extended community filter, a BGP4 extended community filter must be defined.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ip extcommunity-list** command and specify a number to set a BGP extended community filter.

   ```
   device(config-rbridge-id-122)# ip extcommunity-list 1 permit soo 123:2
   device(config)# ip extcommunity-list 1 permit soo 123:2
   ```

4. Enter the **route-map** *name* command to create and define a route map and enter route-map configuration mode.

   ```
   device(config-rbridge-id-122)# route-map extComRmap permit 10
   device(config)# route-map extComRmap permit 10
   ```

   Permits a matching pattern.

5. Enter the **match extcommunity** command and specify an extended community list number.

   ```
   device(config-route-map-extComRmap/permit/10)# match extcommunity 1
   ```

6. Enter the **exit** command.

   ```
   device(config-route-map-extComRmap/permit/10)# exit
   ```

7. Enter the **route-map***name* command to define a route map and enter route-map configuration mode.

   ```
   device(config-rbridge-id-122)# route-map sendExtComRmap permit 10
   device(config)# route-map sendExtComRmap permit 10
   ```

   Permits a matching pattern.

8. Enter the **set extcommunity** command and specify the **rt** *extcommunity value* keyword to specify the route target (RT) extended community attribute.

   ```
   device(config-route-map-sendExtComRmap/permit/10)# set extcommunity rt 3:3
   ```

9. Enter the **set extcommunity** command and specify the **soo** *extcommunity value* keyword to the site of origin (SOO) extended community attribute.

   ```
   device(config-route-map-sendExtComRmap/permit/10)# set extcommunity soo 2:2
   ```

The following example configures an extended community ACL called “extended”, defines a route map, and permits and sets a matching pattern.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip extcommunity-list 1 permit soo 123:2
device(config-rbridge-id-122)# route-map extComRmap permit 10
device(config-route-map-extComRmap/permit/10)# match extcommunity 1
device(config-route-map-extComRmap/permit/1u)# exit
device(config-rbridge-id-122)# route-map sendExtComRmap permit 10
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity rt 3:3
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity soo 2:2

device# configure terminal
device(config)# ip extcommunity-list 1 permit soo 123:2
device(config)# route-map extComRmap permit 10
device(config-route-map-extComRmap/permit/10)# match extcommunity 1
device(config-route-map-extComRmap/permit/1u)# exit
device(config)# route-map sendExtComRmap permit 10
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity rt 3:3
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity soo 2:2
```

Applying a BGP4 extended community filter

A BGP4 extended community filter can be applied.

BGP4 communities must already be defined.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ip extcommunity-list** command and specify a number to set a BGP extended community filter.

   ```
   device(config-rbridge-id-122)# ip extcommunity-list 1 permit rt 123:2
   ```

   ```
   device(config)# ip extcommunity-list 1 permit rt 123:2
   ```

4. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

   ```
   device(config)# router bgp
   ```

5. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

   ```
   device(config-bgp)# local-as 1000
   ```

6. Enter the **neighbor** *ip-address* **remote-as** command to specify the ASN in which the remote neighbor resides.

   ```
   device(config-bgp-router)# neighbor 10.1.2.3 remote-as 1001
   ```

   ```
   device(config-bgp)# neighbor 10.1.2.3 remote-as 1001
   ```

7. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

8. Enter the **neighbor** *ip-address* **activate** command to enable the exchange of information with the neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 activate
```

```
device(config-bgp)# neighbor 10.1.2.3 activate
```

9. Enter the **neighbor** *ip-address* **route-map** command and specify the **in** keyword to apply a route map to incoming routes.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map in extComRmapt
```

```
device(config-bgp)# neighbor 10.1.2.3 route-map in extComRmapt
```

10. Enter the **neighbor** *ip-address* **route-map** command and specify the **out** keyword to apply a route map to outgoing routes.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map out sendExtComRmap
```

```
device(config-bgp)# neighbor 10.1.2.3 route-map out sendExtComRmap
```

11. Enter the **neighbor** *ip-address* **send-community** command and specify the **both** keyword to enable the sending of standard and extended attributes in updates to the specified BGP neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 send-community both
```

```
device(config-bgp)# neighbor 10.1.2.3 send-community both
```

The following example applies a BGP4 extended community filter.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip extcommunity-list 1 permit rt 123:2
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.2.3 remote-as 1001
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 activate
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map in extComRmapt
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map out sendExtComRmap
device(config-bgp-ipv4u)# neighbor 10.1.2.31 send-community both

device# configure terminal
device(config)# ip extcommunity-list 1 permit rt 123:2
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# neighbor 10.1.2.3 remote-as 1001
device(config-bgp)# neighbor 10.1.2.3 activate
device(config-bgp)# neighbor 10.1.2.3 route-map in extComRmapt
device(config-bgp)# neighbor 10.1.2.3 route-map out sendExtComRmap
device(config-bgp)# neighbor 10.1.2.3 send-community both

device# configure terminal
device(config)# ip extcommunity-list 1 permit rt 123:2
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.2.3 remote-as 1001
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 activate
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map in extComRmapt
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map out sendExtComRmap
device(config-bgp-ipv4u)# neighbor 10.1.2.31 send-community both
```

BGP Large Communities

The BGP Large Communities attribute (RFC 8092) supports an optional transitive path attribute of variable length, overcoming previous length limitations. All routes with this attribute belong to the communities specified in the attribute.

Introduction

BGP [RFC 4271] implementations typically support a routing policy language to control the distribution of routing information. Network operators attach BGP communities to routes to associate particular properties with these routes.

A BGP Communities attribute (RFC 1997) is a variable-length attribute consisting of a set of one or more four-octet values, each of which specifies a community. A common use of the individual values of this attribute type splits this single 32-bit value into two 16-bit values. The most significant word is interpreted as an Autonomous System Number (ASN), and the least significant word is a locally defined value whose meaning is assigned by the operator of the AS in the most significant word.

Since the adoption of four-octet ASNs, the BGP Communities attribute can no longer accommodate the above encoding, as a two-octet word cannot fit a four-octet ASN.

To address these shortcomings, a BGP Large Communities attribute is encoded as an unordered set of one or more 12-octet values, each consisting of a four-octet Global Administrator field and two four-octet operator-defined fields, each of which can be used to denote properties or actions significant to the operator of the AS assigning the values.

BGP Large Communities attribute details

Each BGP Large Community value is encoded as a 12-octet quantity, as follows:

```
 0 1 2 3 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Global Administrator                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Local Data Part 1                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Local Data Part 2                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Field	Description
Global Administrator	A four-octet namespace identifier. Allows different ASes to define BGP Large Communities without collision. This field SHOULD be an ASN, in which case the Local Data Parts are to be interpreted as defined by the owner of the ASN.
Local Data Part 1	A four-octet operator-defined value. Represents a function identifier.
Local Data Part 2	A four-octet operator-defined value. Represents a parameter value.

ATTENTION

The use of Reserved ASNs (0 [RFC 7607], 65535, and 4294967295 [RFC 7300] is NOT RECOMMENDED.

There is no significance to the order in which twelve-octet Large Community Attribute values are encoded in a Large Communities attribute, A BGP speaker can transmit them in any order.

In canonical notation, each Large Community value can be summarized as "ASN:Function:Parameter". The function of a community can be divided into two categories for the treatment of routes: Informational Communities and Action Communities.

Informational Communities are labels for attributes such as the origin of the route announcement, the nature of the relation with an External BGP (EBGP) neighbor, or the intended propagation audience.

Action Communities are added as labels to request that a route be treated in a particular way within an AS.

More information on the usage perspective of BGP Large Communities can be obtained from RFC 8195 (Informational RFC) which could serve as an informational reference to real-world applications for BGP Large Communities.

BGP Large Community configuration examples

This section presents a variety of configuration examples.

Defining a large community standard ACL

The following example creates a standard large-community list.

```
device# configure terminal
device(config)# ip large-community-list standard my_std_list seq 10 permit 64497:1:528
```

Defining a large community extended ACL

The following example creates an extended community list with a regular expression.

```
device# configure terminal
device(config)# ip large-community-list extended my_ext_list seq 10 permit 64497:*:*
```

Defining a large community list in a route map

Each route-map instance can have only one set large-community directive in which you can define a maximum of 32 large communities. The following example sets a large-community list in route-map instances to advertise routes with large-community attributes.

```
device# configure terminal
device# configure terminal
device(config)# route-map myroutes permit 10
device(config-route-map-myroutes/permit/10)# set large-community 64497:1:528
device(config-route-map-myroutes/permit/10)# exit
device(config)# route-map myroutes permit 20
device(config-route-map-myroutes/permit/20)# set large-community 64497:5:528
device(config-route-map-myroutes/permit/20)# exit
device(config)# route-map myroutes permit 30
device(config-route-map-myroutes/permit/30)# set large-community 64497:6:528
```

The following example configures multiple large communities for a set large-community directive.

```
device# configure terminal
device(config)# route-map myroutes permit 10
device(config-route-map-myroutes/permit/10)# set large-community 64497:1:528 64497:2:529
```

The following example sets a large-community list in a route-map instance and appends updates to existing attributes.

```
device# configure terminal
device(config)# route-map myroutes permit 10
device(config-route-map-myroutes/permit/10)# set large-community 64497:1:528 additive
```

Defining a route map matching a large-community access list

The following example shows how to configure matching based on a large-community access list named ABCPath for a route map named myroutes.

```
device# config terminal
device(config)# route-map myroutes permit 10
device(config-route-map myroutes/permit/10)# match large-community ABCPath
```

The following example shows how to configure matching based on a large-community access list named lcstdacl1 with an exact match for a route map named myroutes.

```
device# config terminal
device(config)# route-map myroutes permit 10
device(config-route-map myroutes/permit/10)# match large-community lcstdacl1 exact-match
```

Defining a route map to delete large community attributes

The following example deletes community attributes specified in the ACL from the route update. instance.

```
device# configure terminal
device(config)# route-map myroutes permit 10
device(config-route-map-myroutes/permit/10)# set large-community-list mylargecommunitylist delete
```

Enabling the sending of large community attributes in BGP attributes

NOTE

Configure the BGP neighbor before executing the **neighbor send community** command.

If you use the **neighbor send-community** command after a BGP session has been established, the BGP session must be reset for this change to take effect.

The following example sends the large community attributes in BGP attributes.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
(config-bgp-ipv4u)# neighbor 10.11.12.13 send-community large
```

The following example sends all communities.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
(config-bgp-ipv4u)# neighbor 10.11.12.13 send-community all
```

The following example adds the configured route-map for a peer route-map out.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
(config-bgp-ipv4u)# neighbor 10.11.12.13 route-map out myroutes
```

BGP flowspec

BGP flow specification (flowspec) enables a device to instruct BGP neighbors to apply a specific traffic policy or flowspec rule. Neighbors, at their discretion, can apply the policy to take a particular action on traffic or to filter, for example, distributed denial of service (DDoS) attacks.

> **NOTE**
> Currently, only non-VPN IPv4 BGP flowspec is supported (in both default and non-default VRFs).

A traffic policy that an IP router uses to forward traffic, consists of match criteria and corresponding actions. Match criteria operate on different fields in a packet such as source or destination IP prefixes, IP protocol, transport-layer port numbers, and so on. The corresponding actions can be to rate limit, filter, redirect, and so on.

BGP Flowspec is an n-tuple that consists of a number of matching criteria that define an aggregate IP traffic flow specification. The associated set of actions are advertised by using BGP.

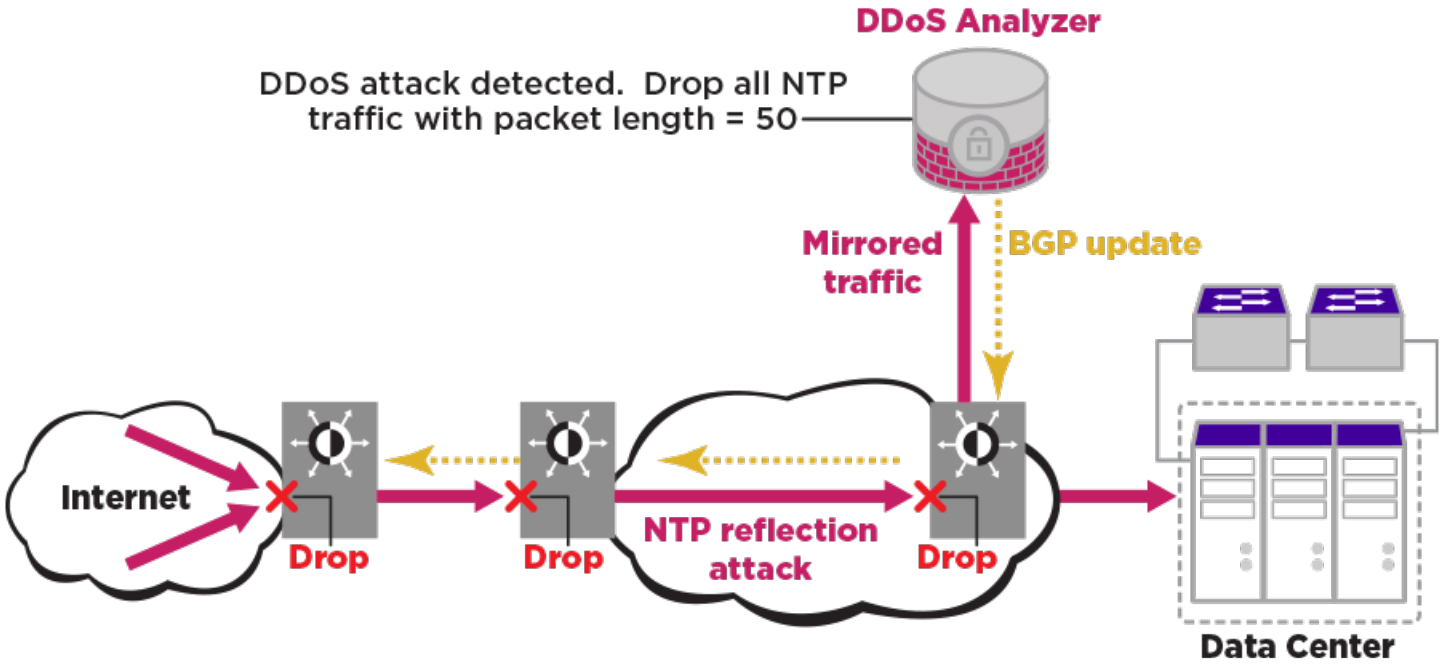
The Extreme Networks implementation of BGP flowspec supports RFC 5575 and conforms to the following standards:

- draft-ietf-idr-bgp-flowspec-oid-06—Revised Validation Procedure for BGP Flow Specifications
- draft-ietf-idr-flowspec-redirect-ip-02—BGP Flow-Spec Redirect to IP Action

BGP flowspec can be applied to DDoS attack filtering. Traditional methods of filtering DDoS attacks include advertising a black hole route to the destination under attack or advertising the destination under attack with a special community that sets the nexthop to a discard nexthop. The disadvantages of traditional approaches include a lack of granularity and the mixing of routing and filtering information. BGP flowspec facilitates a more granular approach to DDoS attack filtering allowing you to propagate policies that filter and police DDoS attacks in a service-provider environment where mitigation points may be in different autonomous systems.

The following figure describes DDoS attack filtering using BGP flowspec.

FIGURE 37 DDoS attack filtering



In the preceding figure, the DDoS Analyzer identifies the threat and generates the filter and corresponding actions according to the flowspec rules. The DDoS appliance then sends out the filter and policing function information as BGP network layer reachability information (NLRI).

Alternatively, by using the CLI, you can manually configure the policy in one of the BGP speakers, such as a route relector, for propagation to other BGP peer devices.

Distribution of flowspec rules by BGP

BGP distributes flow specification rules by using a flow specification Network Layer Reachability Information (NRLI) type.

A BGP application is identified by a specific AFI-SAFI pair. For non-VPN IPv4 BGP flowspec the AFI-SAFI pair is:

- AFI = 1 (IPv4)
- SAFI = 133

The flow specification NRLI type consists of several optional sub-component types. These sub-component types form the n-tuple of the matching criteria. A specific packet is considered to match the flow specification when it matches the all the components types present in the specification. You can define the following sub-component types or tuples.

TABLE 23 BGP flowspec NLRI sub-component types

BGP flowspec NLRI type	Description	Encoding
Type 1	Destination Prefix	<type (1 octet), prefix length (1 octet), prefix>
Type 2	Source Prefix	<type (1 octet), prefix-length (1 octet), prefix>
Type 3	IP Protocol (IPv4) Last Next Header (IPv6)	<type (1 octet), [op, value]+>
Type 4	Port	<type (1 octet), [op, value]+>
Type 5	Destination port	<type (1 octet), [op, value]+>

TABLE 23 BGP flowspec NLRI sub-component types (continued)

BGP flowspec NLRI type	Description	Encoding
Type 6	Source port	<type (1 octet), [op, value]+>
Type 7	ICMP type	<type (1 octet), [op, value]+>
Type 8	ICMP code	<type (1 octet), [op, value]+>
Type 9	TCP flags (CWR, ECE, URG, ACK, PSH, RST, SYN, FIN)	<type (1 octet), [op, bitmask]+>
Type 10	Packet length	<type (1 octet), [op, value]+>
Type 11	DSCP	<type (1 octet), [op, value]+>
Type 12	Fragment (LF, FF, IsF, DF)	<type (1 octet), [op, bitmask]+>

Traffic filtering actions

The default action for filtering BGP flow specification traffic is to accept IP traffic matching the specification.

The following extended community values can be used to specify particular actions for traffic matching the flow specification in the NLRI.

TABLE 24 BGP flowspec traffic filtering actions

Type	Extended community value	Encoding
0x8006	traffic-rate	2-byte as#, 4-byte float
0x8007	traffic-action	bitmask
0x8009	traffic-marking	DSCP value
0x0800 or 0x080C	redirect IPv4	4 byte IPv4

Limitations

When a device is downgraded to a release that is earlier than the 18r.2.00 software release, the BGP flowspec feature does not work.

The current implementation of BGP flowspec has the following limitatons:

Data plane

- Only non-VPN IPv4 BGP flowspec is supported.
- Match types other than the 12 BGP flowspec NLRI sub-componant types described under Distribution of flowspec rules by BGP on page 286 are considered unknown. A flowspec NLRI that contains an unknown match type is considered invalid and is not advertised or installed in the hardware.
- The following TCP flags not supported:
  - Explicit Congestion Notification Echo (ECE)
  - Congestion Window Reduced (CWR)
- Two-byte TCP flags are not supported.
- When a TCP flag sub-componant has a size that is greater than one byte, a RASlog message is triggered and it is not installed in the hardware. However, it is advertised to peer devices.
- In the case of BGP flowspec NLRI sub-componant type 12 (Fragment), only the IsF bit is supported; DF, FF, and LF bit functionality is not supported.
- When the NLRI length needed by match criteria of a stanza in a flowspec route map is greater than 4095, the route map is not installed or advertised by BGP.

- All actions other than the four BGP flowspec traffic filtering actions described in Traffic filtering actions on page 287 are considered unknown. In the case of a flowspec NLRI that contains an unknown action:
  - The unknown action (user-defined extended community or unknown action) is not installed.
  - The remaining flowspec rules are installed.
  - The flowspec NLRI is advertised to peers with the unknown extended communities.
- Copy or mirror action is not supported.
- When a rate-limiting action is set under a BGP flowspec rule, the operational rate value may differ from the rate value specified in the flowspec rule because operational values are selected in multiples of 22 kbits per second.

  NOTE
  When the rate-limiting action under a BGP flowspec rule is set to a value that is lower than 22 kbits per second, matched data traffic is dropped.

- Under the default profile, BGP flowspec routes configured with the following match criteria can only be advertised but not installed in the hardware:
  - IP fragment
  - Packet length
  - ICMP code
  - ICMP type

  To maximise the BGP flowspec match criteria and actions supported in the hardware, a BGP flowspec profile must first be enabled in the hardware by using the **profile tcam bgp-flowspec** command.
- BGP flowspec rules are applied on all Layer 3 interfaces of the specified VRF.
- IPv4 BGP flowspec rules are only applied to IPv4 data traffic; they are not applied to IPv6 data traffic.
- CAM sharing is not supported in the BGP flowspec TCAM profile.
- A number of match commands; for example **match dscp**, support a **range** option. Use the **range** option with caution because a lot of TCAM entries may be created when the rules are expanded.
- Redirection to multiple nexthop addresses is not supported. When multiple redirect nexthops are configured, only the first valid, reachable nexthop is used. In the event that the first nexthop becomes invalid or unreachable, then the next configured valid, reachable nexthop is used.
- Matching of traffic flow with subsequent flowspec rules (terminal-action) is not supported.
- Each flowspec rule may be expanded to several access control list (ACL) rules in TCAM. When TCAM is full a RASlog message is displayed. However, the BGP transit functionality of advertising and receiving flowspec rules can continue when TCAM is full or a flowspec rule is not installed in TCAM.
- Openflow rules are prioritized over BGP flowspec rules. BGP flowspec rules are prioritized over Policy Based Routing (PBR) rules and PBR rules are prioritized over ACL rules.

Control plane

- Extreme recommends a maximum of 1000 (configured and received) flowspec rules.

  NOTE
  In the BGP RIB-in, there is no hardcoded limit for BGP flowspec routes; when sufficient memory exists BGP RIB-in will receive more routes. However, a maximum of 1000 routes is recommended.

- Any match criterion or traffic action in a BGP flowspec route that is not supported in the hardware is still received and advertised by BGP.
- Dampening and soft reconfiguration is not supported for BGP flowspec address family.

- The nexthop path attribute is not added to BGP flowspec routes by default.

Configuration of BGP flowspec

Configuration of BGP flowspec includes optionally configuring a BGP flowspec profile in TCAM, enabling the BGP flowspec address family and address-family neighbors, and then generating and distributing the flowspec rules.

To configure BGP flowspec you must complete the following tasks:

- (Optional) For maximum support of BGP flowspec match criteria and actions, enable a BGP flowspec profile in TCAM by using the **profile tcam bgp-flowspec** command; the default TCAM profile does not support the following flowspec match criteria:
  - Fragment
  - ICMP code
  - ICMP type
  - Packet length
- (Optional) By default, statistics are disabled and when enabled (by using the **ip flowspec rules statistics** command in all relevant VRFs), statistics are only collected for traffic matching BGP flowspec rules. To collect forward or drop statistics, use the **profile counters** command, specifying either the **counter-profile-1** or **counter-profile-4** option.
- Enable the BGP flowspec address family by using the **address-family ipv4 flowspec** command.
- Generate the BGP flowspec rules. Rules are configured as different sequences of a route map.
- Under the BGP flowspec family, enable the neighbors needed to advertise or receive the flowspec rules.
- (Optional) When BGP flowspec validation is not needed, you must disable it at neighbor or peer-group level by using the **neighbor flowspec validation** command, or address-family level by using the **flowspec validation** command.
- (Optional) When the peer device is from another vendor and supports a different implementation of the redirect nexthop action, you must configure the respective interoperational parameters by using the command line interface (CLI) for generating local rules based on the compatability of the peer device.
- Pass the route map to BGP by using the **distribute** command in BGP address-family IPv4 flowspec configuration mode.

Treatment of duplicate stanzas in a BGP flowspec route map

Duplicate stanzas are stanzas with the same match statements. BGP installs only one local flowspec route for a specific key (match statement) and handles duplicate stanzas as follows:

- BGP traverses the flowspec route map in ascending order of sequence number when:
  - The route map is first applied by using the **distribute** command
  - Modifications are made to an active route map.
- While traversing each stanza, BGP attempts to prepare or modify a local flowspec route.
- When the **distribute** command is used to activate a route map containing duplicate stanzas, the stanza with the higher sequence number becomes active only if it has different **set** commands; otherwise the stanza with the lower sequence number remains active. For example, in the case of duplicate stanzas A and B, in which stanza A has a lower sequence number than stanza B, then:
  - When both A and B have same **match** and **set** command content, A is installed, B is not allowed to be active, and a RASlog message is triggered.
  - When both A and B have same **match** command content but different **set** command content, A first becomes active, then B replaces A, and a RASlog message is triggered.
- When any change is made to the **match** content of an active stanza, the stanza with the lowest sequence number from the remaining duplicates is selected as active. For example, in a situation where a route map has three duplicate stanzas (A, B, and

C) in which the **match** command content is the same but the **set** command content differs and where the sequence number of A is lower than the sequence number of B which is lower than the sequence number of C, then, assuming that stanza C is active:

- When the **match** content in stanza C is modified, the stanza with the lowest sequence number among stanza C's former duplicates (A) is selected as active along with the new stanza C and a RASlog message is triggered.
- When stanza C is removed, the stanza with the lowest sequence number among stanza C's former duplicates (A) is selected as active and a RASlog message is triggered.

Enabling a BGP flowspec profile in TCAM

Enabling a BGP flowspec profile in ternary content-addressable memory (TCAM) ensures that the maximum number of BGP flowspec match criteria and actions are supported in the hardware. After the BGP flowspec profile is enabled, the device must be reloaded to activate to the configuration.

When the default TCAM profile is used, the following match criteria can only be advertised but not installed in the hardware:

- IP fragment
- Packet length
- ICMP code
- ICMP type

To enable maximum hardware support for BGP flowspec, perform the following steps to enable a BGP flowspec profile in TCAM.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter hardware configuration mode.

   ```
   device(config)# hardware
   ```

3. Enable a BGP flowspec profile in TCAM

   ```
   device(config-hardware)# profile tcam bgp-flowspec
   ```

4. (Optional) Enable forward or drop statistics.

   ```
   device(config-hardware)# profile counters counter-profile-1
   ```

   You can also enable forward or drop statistics using the **profile counters** command specifying the **counter-profile-4** option. (By default, statistics are disabled and when enabled (by using the **ip flowspec rules statistics** command in all relevant VRFs), statistics are only collected for traffic matching BGP flowspec rules.)

The following example shows how to enable BGP flowspec profile in TCAM and how to enable forward or drop statistics.

```
device# configure terminal
device(config)# hardware
device(config-hardware)# profile tcam bgp-flowspec
device(config-hardware)# profile counters counter-profile-1
```

To activate the configuration, reload the device.

Configuring BGP flowspec rules

BGP flowspec rules consist of match criteria and traffic actions that are configured under a route-map sequence number. Each sequence number or stanza in the route map becomes a flowspec rule.

The following task shows how to configure a selection of BGP flowspec rules. Your particular application may involve configuring a different combination of match criteria and traffic filtering actions.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a route map for BGP flowspec rules that allows a matching pattern.

   ```
   device(config)# route-map flowspec_map permit 56
   ```

   This example creates a route map named flowspec_map that allows a matching pattern and enters configuration mode for the route-map sequence number 56.

   NOTE
   Route-map **deny** sequences are not advertised as flowspec rules; **deny** configurations are ignored.

3. Configure matching based on destination prefix.

   ```
   device(config-route-map-flowspec_map/permit/56)# match ip destination-address 10.2.3.0/24
   ```

4. Configure matching based on protocol number.

   ```
   device(config-route-map-flowspec_map/permit/56)# match protocol eq 30
   ```

5. Configure matching based on a range of protocol numbers.

   ```
   device(config-route-map-flowspec_map/permit/56)# match protocol range 40 50
   ```

6. Configure matching based on traffic having all specified TCP flags.

   ```
   device(config-route-map-flowspec_map/permit/56)# match tcp-flags all not-fin ack
   ```

   In this example, because the **all** option is specified, a match occurs when traffic matches all of the specified options (**not-fin** and **ack**).

7. Configure matching based on traffic having any of the specified TCP flags.

   ```
   device(config-route-map-flowspec_map/permit/56)# match tcp-flags any cwr urg
   ```

   In this example, because the **any** option is specified, a match occurs when traffic matches any of the other specified options (**cwr** or **urg**).

8. Configure matching based on traffic having an ICMP type that is not equal to 5.

   ```
   device(config-route-map-flowspec_map/permit/56)# match ip icmp-type neq 5
   ```

9. Configure a committed information rate (CIR) action.

   ```
   device(config-route-map-flowspec_map/permit/56)# set police cir 0
   ```

10. Configure a mirroring destination.

    ```
    device(config-route-map-flowspec_map/permit/56)# set ip mirror 10.67.67.9
    ```

11. Return to privileged EXEC mode.

```
device(config-route-map-flowspec_map/permit/56)# end
```

The following example shows how to configure a route map for BGP flowspec rules.

```
device# configure terminal
device(config)# route-map flowspec_map permit 56
device(config-route-map-flowspec_map/permit/56)# match ip destination-address 10.2.3.0/24
device(config-route-map-flowspec_map/permit/56)# match protocol eq 30
device(config-route-map-flowspec_map/permit/56)# match protocol range 40 50
device(config-route-map-flowspec_map/permit/56)# match tcp-flags all not-fin ack
device(config-route-map-flowspec_map/permit/56)# match tcp-flags any cwr urg
device(config-route-map-flowspec_map/permit/56)# match ip icmp-type neq 5
device(config-route-map-flowspec_map/permit/56)# set police cir 0
device(config-route-map-flowspec_map/permit/56)# set ip mirror 10.67.67.9
```

When a route map is configured for the first time, BGP flowspec rules are activated by using the **distribute** command in BGP address-family IPv4 flowspec configuration mode.

Enabling BGP flowspec IPv4 address family and activating neighbors

Enabling BGP flowspec IPv4 address family and activating neighbors under this address family supports propogation of a BGP flowspec policy.

Perform the following task to enable BGP flowspec IPv4 address family.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing and enter BGP configuration mode.

```
device(config)# router bgp
```

3. Specify the autonomous system number (ASN) in which the device resides.

```
device(config-bgp-router)# local-as 666
```

4. Specify the ASN in which a remote neighbor resides.

```
device(config-bgp-router)# neighbor 10.61.61.6 remote-as 666
```

5. Enable the IPv4 unicast address family in the VRF.

```
device(config-bgp-router)# address-family ipv4 unicast vrf red
```

This example shows how to enable the IPv4 unicast address family in a VRF instance named red.

> **NOTE**
> The IPv4 unicast address family is always configured on the default VRF.

6. Enable flowspec IPv4 address family.

```
device(config-bgp-router)# address-family ipv4 flowspec vrf red
```

This example shows how to enable IPv4 flowspec address family in a VRF instance named red.

> **NOTE**
> When a VRF instance is not specified, IPv4 flowspec address family is enabled in the default VRF.

7. Activate flowspec route exchange with the remote neighbor under the BGP flowspec IPv4 address family.

```
device(config-bgp-ipv4fs)# neighbor 10.61.61.6 activate
```

8. (Optional) When the receiving device supports the **0x0800** extended-community type or the **nlri** holder (place where the nexthop IP address is encoded in the BGP update packet) or both, configure the flowspec redirect holder or nexthop action type for the neighbor.

```
device(config-bgp-ipv4fs)# neighbor 10.61.61.6 flowspec redirect holder nrli next-hop type 0x0800
```

The following example shows how to configure an IPv4 unicast address family in a VRF named red, enable BGP flowspec IPv4 address family in the VRF, activate a neighbor (10.61.61.6) under under BGP flowspec IPv4 address family and configure both the flowspec redirect holder and nexthop action type for the neighbor.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 666
device(config-bgp-router)# neighbor 10.61.61.6 remote-as 666
device(config-bgp-router)# address-family ipv4 unicast vrf red
device(config-bgp-router)# address-family ipv4 flowspec vrf red
device(config-bgp-ipv4fs)# neighbor 10.61.61.6 activate
device(config-bgp-ipv4fs)# neighbor 10.61.61.6 flowspec redirect holder nlri next-hop type 0x0800
```

When you have activated all the neighbors needed under BGP flowspec IPv4 address family, configure distribution of BGP flowspec rules by using the **distribute** command in BGP address-family IPv4 flowspec configuration mode.

Distributing BGP flowspec rules

Flowspec rules are distributed by BGP when a route map that is configured with flowspec rules is passed to BGP for advertisement by BGP neighbors.

Before the route map is passed to BGP:

- The BGP flowspec rules must be configured under the route map.
- The BGP flowspec IPv4 address family must be configured on the device.
- The neighbors needed to advertise the flowspec rules must be enabled under the BGP flowspec IPv4 address family.

There is an example at the end of this task that shows all the configuration steps in order.

Perform the following steps to distribute BGP flowspec rules.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP and enter BGP configuration mode.

```
device(config)# router bgp
```

3. Enter BGP address-family IPv4 flowspec configuration mode.

```
device(config-bgp-router)# address-family ipv4 flowspec
```

4. Distribute a route map that is already configured with the BGP flowspec rules.

```
device(config-bgp-ipv4fs)# distribute flowspec-map
```

This example shows how to distribute a route map named flowspec-map under the BGP flowspec IPv4 address family.

The following example shows how to configure a route map named flowspec-map with BGP flowspec rules, enable the BGP flowspec IPv4 address family on the device, activate a neighbor under the BGP flowspec IPv4 address family, and distribute the route map.

```
device# configure terminal
device(config)# route-map flowspec_map permit 56
device(config-route-map-flowspec_map/permit/56)# match ipv4 destination-address 10.2.3.0/24
device(config-route-map-flowspec_map/permit/56)# match protocol eq 30
device(config-route-map-flowspec_map/permit/56)# match protocol range 40 50
device(config-route-map-flowspec_map/permit/56)# match tcp-flags all not-fin ack
device(config-route-map-flowspec_map/permit/56)# match tcp-flags any cwr urg
device(config-route-map-flowspec_map/permit/56)# match ip icmp-type neq 5
device(config-route-map-flowspec_map/permit/56)# set police cir 0
device(config-route-map-flowspec_map/permit/56)# set ip mirror 10.67.67.9
device(config-route-map-flowspec_map/permit/56)# end
!
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 666
device(config-bgp-router)# neighbor 10.61.61.6 remote-as 666
device(config-bgp-router)# address-family ipv4 unicast vrf red
device(config-bgp-router)# address-family ipv4 flowspec vrf red
!
device(config-bgp-ipv4fs)# neighbor 10.61.61.6 activate
device(config-bgp-ipv4u)# exit
!
device(config-bgp-ipv4fs)# distribute flowspec-map
```

Updating a previously-distributed route map

To generate less installation requests in the hardware when you make multiple updates to a previously-distrubuted BGP flowspec route map, you can configure a delay period before application of the updates.

Before completing this task, it is assumed that a BGP flowspec route map has already been created and distributed.

By default, updates to a previously-distributed BGP flowspec route map are applied after a delay of 10 seconds. To send a lesser number of changes to the hardware when making multiple updates, you can configure a longer delay period by using the **filter-change-update-delay** command. The following task shows how to configure a delay period of 500 seconds and updates a selection of rules for a previously-distributed route map for BGP flowspec. Your particular application may involve updating a different combination of match criteria and traffic filtering actions.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Configure the delay period that should elapse before application of changes made to BGP flowspec route-map rules.

```
device(config)# filter-change-update-delay 500
```

This example configures a delay period of 500 seconds. By default, changes to BGP flowspec rules are applied after 10 seconds.

3. Enter configuration mode for a sequence number in a previously-distributed BGP flowspec route map.

```
device(config)# route-map flowspec_map permit 56
```

   This example enters configuration mode for sequence number 56 under a route map named flowspec_map.

4. Configure matching based on source address.

```
device(config-route-map-flowspec_map/permit/56)# match ipv4 source-address 10.3.2.0/24
```

5. Configure matching based on port number.

   The following example specifies that matching occurs when the source or destination port number is less than 40.

```
device(config-route-map-flowspec_map/permit/56)# match port lt 40
```

   The following example specifies that matching occurs when the source or destination port number is not equal to 30.

```
device(config-route-map-flowspec_map/permit/56)# match port neq 30 67 89
```

   When both of the above configurations are applied, matching occurs when the source or destination port number is less than 40 or is not equal to 30 and 67 and 89.

6. Configure matching based on fragment type.

   The following example specifies that matching occurs when the traffic fragment type is **first-fragment**.

```
device(config-route-map-flowspec_map/permit/56)# match fragment-type all first-fragment
```

   The following example specifies that matching occurs when the traffic fragment type is either **dont-fragment** or **not-last-fragment**.

```
device(config-route-map-flowspec_map/permit/56)# match fragment-type any dont-fragment not-last-
fragment
```

   When both of the above configurations are applied, matching occurs when the fragment type is equal to **first-fragment**, or **dont-fragment**, or **not-last-fragment**.

7. Enable traffic sampling.

```
device(config-route-map-flowspec_map/permit/56)# set sflow
```

8. Configure a nexthop address.

```
device(config-route-map-flowspec_map/permit/56)# set ip nexthop 10.89.89.7
```

9. Return to privileged EXEC mode.

```
device(config-route-map-flowspec_map/permit/56)# end
```

10. Install the new rules in the hardware by clearing the delay time.

```
device# clear filter-change-update
```

   The **clear filter-change-update** command clears the previously-configured delay time of 500 seconds. This prevents unnecessary delay in installing the changes in the hardware.

The following example shows how configure a delay (500 seconds) for the application of updates to a BGP flowspec route-map rule, make a selection of rule updates, and then install the updated rules in the hardware by clearing the delay time.

```
device# configure terminal
device(config)# route-map flowspec_map permit 56# filter-change-update-delay 500
device(config)# route-map flowspec_map permit 56# route-map flowspec_map permit 56
device(config-route-map-flowspec_map/permit/56)# match ipv4 source-address 10.3.2.0/24
device(config-route-map-flowspec_map/permit/56)# match port neq 30 67 89
device(config-route-map-flowspec_map/permit/56)# match port lt 40
device(config-route-map-flowspec_map/permit/56)# match fragment-type all first-fragment
device(config-route-map-flowspec_map/permit/56)# match fragment-type any dont-fragment not-last-fragment
device(config-route-map-flowspec_map/permit/56)# set set sflow
device(config-route-map-flowspec_map/permit/56)# set ip nexthop 10.89.89.7
device(config-route-map-flowspec_map/permit/56)# end
device# clear filter-change-update
```

Enabling statistics for BGP flowspec rules

By default, BGP flowspec statistics are disabled.

Perform the following steps to enable BGP flowspec statistics.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable BGP flowspec statistics.

   ```
   device(config)# ip flowspec rules statistics
   ```

3. Return to global configuration mode.

   ```
   device(config)# exit
   ```

4. Statistics for BGP flowspec can be displayed by using the **show ip flowspec rules detail** command.

   ```
   device# show ip flowspec rules detail

   VRF :default-vrf
   Total number of Rules: 1

   1 Origin: Local(flowmap:23) Active: Yes
     Match:
        DSCP         <60
     Actions:
        Traffic-rate asn 666, rate 125000 bits/sec(operational-rate 132000 bits/sec)

   Statistic         packets/bytes
   -------------     ------------
      Matched           17412786/12589441782
      Transmitted       1453/1048023
      dropped           17411333/12588393759
   ```

The following example shows how to enable BGP flowspec statistics.

```
device# configure terminal
device(config)# ip flowspec rules statistics
```

Displaying and clearing statistics for IPv4 flowspec rules

You can display and clear statistical information for IPv4 flowspec rules.

Before displaying or clearing statistical information for IPv4 flowspec rules, you should enable the collection of statistical information by using the **ip flowspec rules statistics** command. There is a configuration example at the end of this task that shows all the configuration steps in order.

Perform the following task to display and then clear statistics for IPv4 flowspec rules.

1. From privileged EXEC mode, display statistics for IPv4 flowspec rules. The following example displays detailed statistical information.

```
device# show ip flowspec rules detail

VRF :default-vrf
Total number of Rules: 1

1 Origin: Local(flowmap:23) Active: Yes
  Match:
     DSCP         <60
  Actions:
     Traffic-rate asn 666,rate 125000 bits/sec(operational-rate 132000 bits/sec)

  Statistic       packets/bytes
  -------------     ------------
    Matched         17412786/12589441782
    Transmitted     1453/1048023
    dropped         17411333/12588393759
```

2. Clear statistics for IPv4 flowspec rules.

```
device# clear ip flowspec rules statistics
```

3. Verify that statistics for IPv4 flowspec rules are cleared.

```
device# show ip flowspec rules detail

VRF :default-vrf
Total number of Rules: 1

1 Origin: Local(flowmap:23) Active: Yes
  Match:
     DSCP         <60
  Actions:
     Traffic-rate asn 666,rate 125000 bits/sec(operational-rate 132000 bits/sec)

  Statistic       packets/bytes
  -------------     ------------
    Matched         0/0
    Transmitted     0/0
    dropped         0/0
```

The following example shows how to enable BGP flowspec rules, display statistics for BGP flowspec rules, clear the statistical information and then verify that the statistics have been cleared.

```
device# configure terminal
device(config)# ip flowspec rules statistics
device(config)# exit
!
device# show ip flowspec rules detail

VRF :default-vrf
Total number of Rules: 1

1 Origin: Local(flowmap:23) Active: Yes
  Match:
     DSCP         <60
  Actions:
     Traffic-rate asn 666,rate 125000 bits/sec(operational-rate 132000 bits/sec)

  Statistic       packets/bytes
  -------------     ------------
    Matched         17412786/12589441782
    Transmitted     1453/1048023
    dropped         17411333/12588393759


!
device# clear ip flowspec rules statistics
!
device# show ip flowspec rules detail

VRF :default-vrf
Total number of Rules: 1

1 Origin: Local(flowmap:23) Active: Yes
  Match:
     DSCP         <60
  Actions:
     Traffic-rate asn 666,rate 125000 bits/sec(operational-rate 132000 bits/sec)

  Statistic       packets/bytes
  -------------     ------------
    Matched         0/0
    Transmitted     0/0
    dropped         0/0
```

Configuration example for BGP flowspec

BGP flowspec configuration tasks include enabling a BGP flowspec profile in ternary content-assressable memory (TCAM), configuring a flowspec policy (match criteria and traffic actions), enabling a BGP flowspec IPv4 address family and address-family neighbors, and distributing the flowspec policy. The following configuration example shows all the steps in sequence.

Because the default TCAM profile does support flowspec match criteria such as Fragment, ICMP code, ICMP type, and Packet length, you must enable a BGP flowspec profile when you want to maximise BGP flowspec support in the hardware. The following example shows how to enable a BGP flowspec profile in TCAM.

```
device# configure terminal
device(config)# hardware
device(config-hardware)# profile tcam bgp-flowspec
```

> NOTE
> After configuration, the device must be reloaded.

By default, statistics are disabled and when enabled (by using the **ip flowspec rules statistics** command in all relevant VRFs), statistics are only collected for traffic matching BGP flowspec rules. The following example enables collection of forward or drop statistics.

```
device(config-hardware)# profile counters counter-profile-1
device(config-hardware)# end
```

The following example shows how to enable statistics.

```
device# configure terminal
device(config)# ip flowspec rules statistics
```

The following example sets BGP flowspec rules for sequence number 56 under a route map named flowspec_map.

```
device(config)# route-map flowspec_map permit 56
device(config-route-map-flowspec_map/permit/56)# match ipv4 destination-address 10.2.3.0/24
device(config-route-map-flowspec_map/permit/56)# match protocol eq 30
device(config-route-map-flowspec_map/permit/56)# match protocol range 40 50
device(config-route-map-flowspec_map/permit/56)# match tcp-flags all not-fin ack
device(config-route-map-flowspec_map/permit/56)# match tcp-flags any cwr urg
device(config-route-map-flowspec_map/permit/56)# match ip icmp-type neq 5
device(config-route-map-flowspec_map/permit/56)# set police cir 0
device(config-route-map-flowspec_map/permit/56)# set ip mirror 10.67.67.9
device(config-route-map-flowspec_map/permit/56)# exit
```

The following example enables a BGP flowspec IPv4 address family and activate an address-family neighbor.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 666
device(config-bgp-router)# neighbor 10.61.61.6 remote-as 666
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# no neighbor 10.61.61.6 activate
device(config-bgp-ipv4u)# exit
!
device(config-bgp-router)# address-family ipv4 flowspec
device(config-bgp-ipv4fs)# neighbor 10.61.61.6 activate
```

By default, the redirect nexthop action **type** option for a neighbor is set to **0x0800c**, while the **holder** option (place where the nexthop IP address is encoded in the BGP update packet) is set to **extended-community**. You can configure alternate values when the receiving device supports the **0x0800** extended-community type or the **nlri** holder. The following example sets the redirect nexthop action type to **0x0800** for a neighbor (10.61.61.1).

```
device(config-bgp-ipv4fs)# neighbor 10.61.61.6 flowspec redirect next-hop type 0x0800
device(config-bgp-ipv4fs)# exit
```

The following example distributes the route map named flowspec-map to BGP.

```
device(config-bgp-router)# address-family ipv4 flowspec
device(config-bgp-ipv4fs)# distribute flowspec-map
device# end
```

When there is a subsequent need to update the BGP flowspec rules, configuring a delay period before updating the rules reduces the number of installation requests in the hardware. The following example shows how to configure a delay period and some rule updates.

```
device# configure terminal
device(config)# route-map flowspec_map permit 56# filter-change-update-delay 500
device(config)# route-map flowspec_map permit 56# route-map flowspec_map permit 56
device(config-route-map-flowspec_map/permit/56)# match ipv4 source-address 10.3.2.0/24
device(config-route-map-flowspec_map/permit/56)# match port neq 30 67 89
device(config-route-map-flowspec_map/permit/56)# match port lt 40
device(config-route-map-flowspec_map/permit/56)# match fragment-type all first-fragment
device(config-route-map-flowspec_map/permit/56)# match fragment-type any dont-fragment not-last-fragment
device(config-route-map-flowspec_map/permit/56)# set set sflow
device(config-route-map-flowspec_map/permit/56)# set ip nexthop 10.89.89.7
device(config-route-map-flowspec_map/permit/56)# end
device# clear filter-change-update
```

BGP4 graceful restart

BGP4 graceful restart (GR) allows for restarts where BGP neighboring devices participate in the restart, helping to ensure that no route and topology changes occur in the network for the duration of the restart.

The GR feature provides a routing device with the capability to inform its neighbors when it is performing a restart.

> NOTE
> Process restart takes precedence over graceful restart.

When a BGP session is established, GR capability for BGP is negotiated by neighbors through the BGP OPEN message. If the neighbor also advertises support for GR, GR is activated for that neighbor session. If neither peer exchanges the GR capability, the session is not GR-capable. If the BGP session is lost, the BGP peer router, known as a GR helper, marks all routes associated with the device as "stale" but continues to forward packets to these routes for a set period of time. The restarting device also continues to forward packets for the duration of the graceful restart. When the graceful restart is complete, routes are obtained from the helper so that the device is able to quickly resume full operation.

When the GR feature is configured on a device, both helper router and restarting router functionalities are supported. It is not possible to disable helper functionality explicitly.

GR is disabled by default and can be enabled for both IPv4 and IPv6 address families. When the GR timer expires, the BGP RASlog message is triggered.

> NOTE
> BGP4 GR can be configured for a global routing instance or for a specified VRF instance.

> NOTE
> BGP4 GR is fully supported by MLX Series and XMR Series devices. The CER 2000 Series and CES 2000 Series devices only support GR helper mode.

Configuring BGP4 graceful restart

The graceful restart (GR) feature can be configured on a routing device, providing it with the capability to inform its neighbors when it is performing a restart.

> NOTE
> High availability (HA) requires GR to be enabled.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
device(config)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
device(config-bgp)# local-as 1000
```

5. Enter the **neighbor** *ip address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp)# neighbor 10.11.12.13 remote-as 2
```

6. Enter the **address-family ipv4 unicast** command to enter IPv4 address-family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

7. Enter the **graceful-restart** command to enable the graceful restart feature.

```
device(config-bgp-ipv4u)# graceful-restart
device(config-bgp)# neighbor 10.11.12.13 remote-as 2
```

8. Do any of the following:
   - Enter the **graceful-restart** command and use **purge-time** parameter to overwrite the default purge-time value.

     ```
     device(config-bgp-ipv4u)# graceful-restart purge-time 300
     device(config-bgp)# graceful-restart purge-time 300
     ```

   - Enter the **graceful-restart** command and use **restart-time** parameter to overwrite the default restart-time advertised to graceful restart-capable neighbors.

     ```
     device(config-bgp-ipv4u)# graceful-restart restart-time 180
     device(config-bgp)# neighbor graceful-restart restart-time 180
     ```

   - Enter the **graceful-restart** command and use **stale-routes-time** parameter to overwrite the default amount of time that a helper device will wait for an EOR message from a peer.

     ```
     device(config-bgp-ipv4u)# graceful-restart stale-routes-time 100
     device(config-bgp)# graceful-restart stale-routes-time 100
     ```

The following example enables the GR feature.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1
device(config-bgp)# neighbor 10.11.12.13 remote-as 2
device(config-bgp)# graceful-restart

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# graceful-restart

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# graceful-restart
```

The following example enables the GR feature and sets the purge time to 300 seconds, over-writing the default value.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1
device(config-bgp)# neighbor 10.11.12.13 remote-as 2
device(config-bgp)# graceful-restart purge-time 180

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# graceful-restart purge-time 180

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# graceful-restart purge-time 180
```

The following example enables the GR feature and sets the restart time to 180 seconds, over-writing the default value.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1
device(config-bgp)# neighbor 10.11.12.13 remote-as 2
device(config-bgp)# graceful-restart restart-time 180

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# graceful-restart restart-time 180

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# graceful-restart restart-time 180
```

The following example enables the GR feature and sets the stale-routes time to 100 seconds, over-writing the default value.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1
device(config-bgp)# neighbor 10.11.12.13 remote-as 2
device(config-bgp)# graceful-restart stale-routes-time 100

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# graceful-restart stale-routes-time 100

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# graceful-restart stale-routes-time 100
```

Use the **clear ip bgp neighbor** command with the **all** parameter for the changes to the GR parameters to take effect immediately.

BGP dynamic neighbors

The BGP dynamic neighbors feature allows peering to a group of remote neighbors defined by a listen range. BGP neighbors can be created without statically configuring them.

Routing protocols are designed to exchange routing information and distribute it to all neighbors. Each protocol has its own way of detecting neighbors and establishing adjacency with them. By design, all IGP neighbors are a single hop away. However, this is not true for BGP. BGP neighbors can be a single hop away or *n* hops away. This implies that BGP neighbors are not dynamically discovered. Rather, an administrator must enable and configure each neighbor for the exchange of routing information. The greater the number of neighbors, the greater the administrative overhead. With BGP dynamic neighbors, when a device sees an incoming TCP session initiated from a remote neighbor with an IP address in the configured subnet listen range, a new BGP neighbor is dynamically created. The device that initiates this connection still has the neighbor statically configured. Each listen range can be configured as a subnet IP range. BGP dynamic neighbors are configured using a range of IP addresses and BGP peer groups.

Whenever the incoming TCP session falls under the configured subnet listen range, a new BGP neighbor is dynamically created as a member of that peer group. Once the listen range is configured, and the associated peer group is activated, dynamic BGP neighbor creation does not require any further configuration on the initial device. Newly created BGP dynamic peers automatically inherit the attributes associated with the peer group attached to the listen range, such as inbound policy, outbound policy, and so on.

You can set the total number of dynamic neighbors that can be formed in the system. When the global or listen range limit is increased, any new connection coming from the remote end that falls under the configured range is accepted. If the limit has already been reached and you reduce the global or peer-group limit, already established dynamic neighbors are not destroyed. Once the limit has been reached and new connection requests are received, the connections are rejected and the information is logged.

BGP dynamic neighbors are deleted when a session moves from the established state to the idle state.

> NOTE
> BGP dynamic neighbors supports only IPv4 BGP peering.

Consider the following when using BGP dynamic neighbors:

- A neighbor can be associated with a peer group so that the peer group properties are inherited by each individual neighbor.
- Individual dynamic neighbors cannot be configured with a separate set of policies as BGP dynamic neighbors are created on demand.
- BFD for dynamic neighbors is supported. When Bidirectional Forwarding Detection (BFD) detects that a link has gone down, the dynamic neighbor moves back to the idle state and the dynamic neighbor is deleted.
- Static neighbors are always given precedence over BGP dynamic neighbors.
- The **auto-shutdown-new neighbors** command is not supported for BGP dynamic neighbors.
- BTSH and MD5 passwords are not supported for BGP dynamic neighbors.

Graceful restart for BGP dynamic neighbors

Consider the following when configuring graceful restart (GR) for dynamic neighbors.

- A dynamic neighbor negotiates the graceful restart (GR) capability with its neighbors in the same way as a statically configured neighbor. Dynamic neighbors are re-established after a warm reboot. The newly formed neighbors follow the same restart protocol as that of statically configured neighbors. Dynamically created neighbors initiate the TCP connect after a GR. However, as is not the case with static neighbors, if the dynamic neighbors do not come up within the configured restart time they are deleted. If a dynamic neighbor does come up, but fails to receive the EOR within the configured purge time, all the received routes are deleted. However, the dynamic neighbors are not deleted.

- When a dynamic neighbor moves from the established state to the idle state, all the routes are cleaned up and the neighbor is deleted. Deleting the neighbor breaks the existing GR protocol. If a dynamic neighbor has negotiated the GR and the session is moving to idle, the neighbor is deleted.

  Like static neighbors, GR helper marks all routes received from the restarting node as stale and starts the restarting timer. If the restarting node does not come within this restart timer period, all routes marked as STALE are deleted, followed by the neighbor deletion. If the restarting node comes up but fails to send the EOR, all stale routes are deleted, but the neighbor are not deleted.

Configuring BGP dynamic neighbors

BGP dynamic neighbors can be configured. The following task sets a global limit on the number of dynamic BGP neighbors, creates a BGP peer group. and associates a subnet range with the peer group. The peer group is added to the BGP neighbor table of the local device and an alternate ASN is configured. For information on configuring BGP dynamic neighbors for an IP fabric, refer to the *IP Fabrics Configuration Guide*.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   device(config)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   ```

5. Enter the **neighbor peer-group** command, specifying a name for the peer group, to create a peer-group.

   ```
   device(config-bgp-router)# neighbor mypeergroup1 peer-group
   ```

6. Enter the **neighbor** *peer-group* **remote-as** command to specify an AS for the peer-group.

   ```
   device(config-bgp-router)# neighbor mypeergroup1 remote-as 80
   ```

7. Enter the **listen-range** command, specifying an IP address and mask, to associate a subnet range with the BGP peer group. Use the **limit** parameter and specify a value to set the maximum number of BGP dynamic neighbors that can be created.

   ```
   device(config-bgp-router)# listen-range 150.1.0.0/16 peer-group mypeergroup1 limit 20
   ```

8. Enter the **neighbor alternate-as** command with the **add** parameter, specifying a value, to set an alternate AS number for listen range neighbors in the configured peer group.

   ```
   device(config-bgp-router)# neighbor mypeergroup1 alternate-as add 101
   ```

The following example sets a global limit on the number of dynamic BGP neighbors, creates a BGP peer group, and associates a subnet range with the peer group. The peer group is added to the BGP neighbor table of the local device and an alternate ASN is configured.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor mypeergroup1 peer-group
device(config-bgp-router)# neighbor mypeergroup1 remote-as 80
device(config-bgp-router)# listen-range 150.1.0.0/16 peer-group mypeergroup1 limit 20
device(config-bgp-router)# neighbor mypeergroup1 alternate-as add 101
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor mypeergroup1 peer-group
device(config-bgp-router)# neighbor mypeergroup1 remote-as 80
device(config-bgp-router)# listen-range 150.1.0.0/16 peer-group mypeergroup1 limit 20
device(config-bgp-router)# neighbor mypeergroup1 alternate-as add 101
```

BGP add path overview

The BGP add path feature provides the ability for multiple paths for the same prefix to be advertised without the new paths implicitly replacing the previous paths. Path diversity is achieved rather than path hiding.

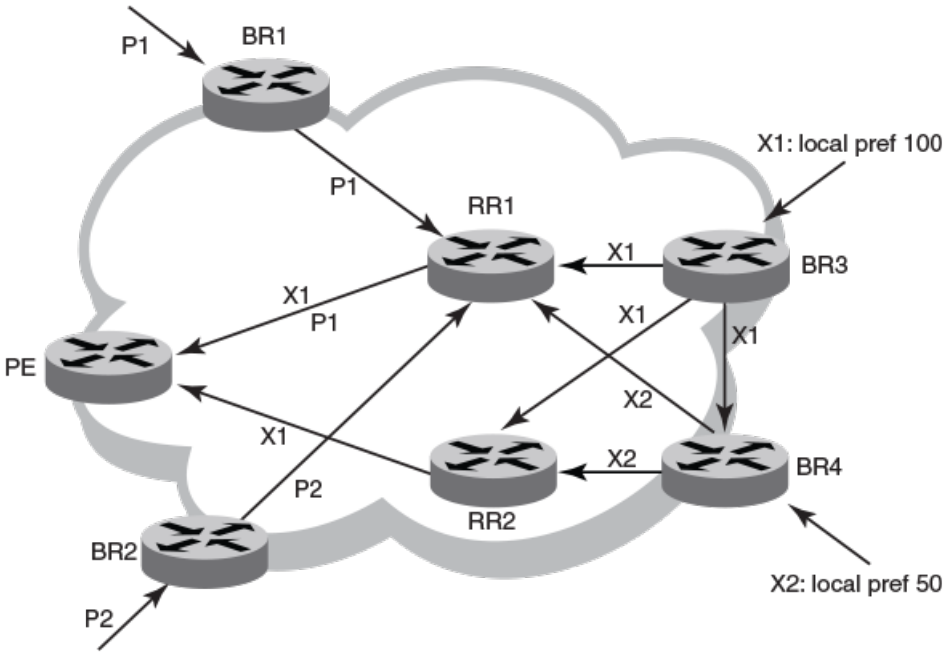
BGP devices and route reflectors (RRs) generally advertise only their best path to neighboring devices, even when multiple paths exist. The advertisement of the same prefix from the same neighbor replaces the previous announcement of that prefix. This is known as an implicit withdraw, behavior that achieves better scaling but at the cost of path diversity. When RRs are deployed inside an autonomous system (AS) to avoid iBGP scaling issues, only the best route is reflected even if multiple Equal-Cost Multipath (ECMP) routes exist. These secondary best paths remain hidden.

Path hiding can affect the efficient use of BGP multipath and path diversity, and prevent hitless planned maintenance. Upon next hop failures, path hiding also inhibits fast and local recovery because the network has to wait for BGP control plane convergence to restore traffic. BGP add path enables BGP to advertise even the secondary best routes so that multiple paths for the same prefix can be advertised without the new paths implicitly replacing previous paths. BGP add path provides a generic way of offering path diversity.

In the figure below, path hiding occurs in two ways:

- Prefix P has paths P1 and P2 advertised from BR1 and BR2 to RR1. RR1 selects P1 as the best path and advertises only P1 to PE.
- Prefix X has path X1 advertised from BR3 to BR4 with local preference 100. BR4 also has path X2. However, only the best path, X1, is selected. BR3 advertises X1 to the RRs and X2 is suppressed.

FIGURE 38 BGP path hiding



NOTE
BGP add path is supported for both route reflectors and confederations.

NOTE
BGP add path is supported for both VCS fabrics and IP fabrics, but is supported for the IPv4 and IPv6 unicast address families only.

Advantages and limitations of BGP add path

The following are the advantages and limitations provided by BGP add path.

Advantages of BGP add path

- **Fast convergence and fault tolerance**: When BGP add path is enabled, more than one path to a destination is advertised. If one of the paths goes down, connectivity is easily restored due to the availability of backup paths. If the next hop for the prefix becomes unreachable, the device can switch to the backup route immediately without having to wait for BGP control plane messages.
- **Enhanced load balancing capabilities**: Traditionally with RRs in an iBGP domain, only the best path is given to the clients even if ECMP paths exists. This affects load balancing. With additional paths advertised by RRs, the clients have more effective load balance.

Limitations of BGP add path

- Load balancing is achieved for the BGP IPv4 and IPv6 address family neighbors only and not for EVPN neighbors. For more information on BGP EVPN, refer to the BGP EVPN chapter.

- BGP add path does not provide any advantage for ARP and MAC routes supported by EVPN peers. L2 ECMP is not currently supported.
- BGP multipath must be configured to choose ECMP paths. Only the best path and the first 5 ECMP paths are chosen as additional paths for basic functionality support and advertised to neighbors with path IDs.

BGP add path functionality

BGP add path is implemented by including an additional four octet value known as a path identifier (ID) for each path in the NLRI. Path IDs are unique to a peering session and are generated for each network.

A path ID can apply to the IPv4 or IPv6 unicast address family.

Therefore, when the same prefix is received with the same path ID from the same peer, it is considered as a replacement route or a duplicate route. When the same prefix is received with a different path ID from the same peer, it is considered as an additional path to the prefix.

To send or receive additional paths, the add path capability must be negotiated. If it isn't negotiated, only the best path can be sent. BGP updates carry the path ID once the add path capability is negotiated. In order to carry the path ID in an update message, the existing NLRI encodings are extended by prepending the path ID field, which consists of four octets.

The assignment of the path ID for a path by a BGP device occurs in such a way that the BGP device is able to use the prefix and path ID to uniquely identify a path advertised to a neighbor so as to continue to send further updates for that path. The receiving BGP neighbor that re-advertises a route regenerates its own path ID to be associated with the re-advertised route.

The set of additional paths advertised to each neighbor can be different, and advertise filters are provided on a per neighbor basis.

Negotiating BGP4 add paths capability

BGP4 neighbors can be configured to send or receive additional paths after negotiation is completed.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   ```

5. Enter the **neighbor** *ipv4 address* **remote-as** command to add a neighbor.

   ```
   device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
   ```

6. Enter the **address-family unicast** command using the **ipv4** parameter to enter BGP address-family IPv4 unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

7. Enter the **neighbor capability additional-paths** command, specifying an IP address, to enable the advertisement of additional paths. Enter the **send** parameter to enable BGP4 to send additional paths to BGP neighbors.

```
device(config-bgp-ipv4u)# neighbor 10.10.10.1 capability additional-paths send
```

8. Enter the **neighbor capability additional-paths** command, specifying an IP address, to enable the advertisement of additional paths. Enter the **receive** parameter to enable BGP4 toreceive additional paths from BGP neighbors.

```
device(config-bgp-ipv4u)# neighbor 10.10.10.1 capability additional-paths receive
```

The following example enables BGP4 to receive additional paths from BGP neighbors and to send additional paths to BGP neighbors.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.10.10.1 capability additional-paths send
device(config-bgp-ipv4u)# neighbor 10.10.10.1 capability additional-paths receive
```

Advertising best BGP4 additional paths

Additional paths that the device selects as best paths can be advertised.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **neighbor** *ipv4 address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
```

6. Enter the **address-family unicast** command using the **ipv4** parameter to enter BGP address-family IPv4 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

7. Enter the **neighbor additional-paths advertise** command, specifying an IP address, using the **best** *value* parameter to configure BGP4 to advertise the best BGP path and an additional four routes to this neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.10.10.1 additional-paths advertise best 4
```

The following example configures BGP4 to advertise the best BGP path and an additional four routes a to a BGP neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-router)# neighbor 10.10.10.1 additional-paths advertise best 4
```

Advertising all BGP4 additional paths

Additional paths can be advertised.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   ```

5. Enter the **neighbor** *ipv4 address* **remote-as** command to add a neighbor.

   ```
   device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
   ```

6. Enter the **address-family unicast** command using the **ipv4** parameter to enter BGP address-family IPv4 unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

7. Enter the **neighbor additional-paths advertise** command, specifying an IP address, using the **all** parameter to configure BGP4 to advertise all BGP4 paths with a unique next hop.

   ```
   device(config-bgp-ipv4u)# neighbor 10.10.10.1 additional-paths advertise all
   ```

The following example configures BGP4 to advertise all BGP additional paths.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.10.10.1 additional-paths advertise all
```

BGP additional-paths overview

BGP additional-paths provides the ability for multiple paths for the same prefix to be advertised without the new paths implicitly replacing the previous paths. Path diversity is achieved rather than path hiding.

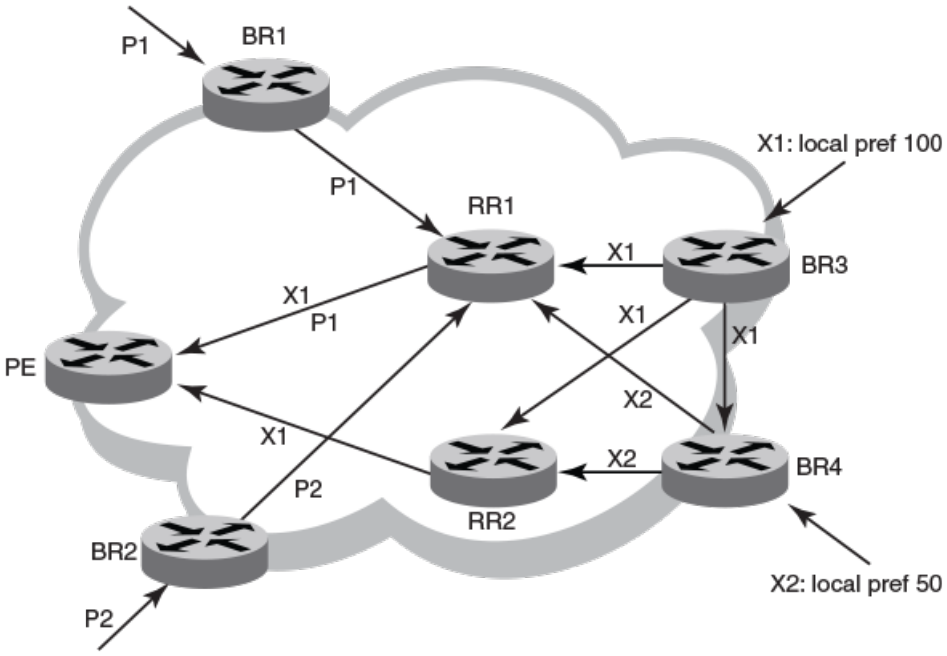
BGP devices generally advertise only their best path to neighboring devices, even when multiple paths exist. The advertisement of the same prefix from the same neighbor replaces the previous announcement of that prefix. This is known as an implicit withdraw, behavior that achieves better scaling but at the cost of path diversity.

Path hiding can affect the efficient use of BGP multipath and path diversity, and prevent hitless planned maintenance. Upon next hop failures, path hiding also inhibits fast and local recovery because the network must wait for BGP control plane convergence to restore traffic. BGP additional-paths enables BGP to advertise even the secondary best routes so that multiple paths for the same prefix can be advertised without the new paths implicitly replacing previous paths. BGP additional-paths provides a generic way of offering path diversity.

In the following figure, path hiding occurs in two ways:

- Prefix P has paths P1 and P2 advertised from BR1 and BR2 to RR1. RR1 selects P1 as the best path and advertises only P1 to PE.
- Prefix X has path X1 advertised from BR3 to BR4 with local preference 100. BR4 also has path X2. However, only the best path, X1, is selected. BR3 advertises X1 to the RRs and X2 is suppressed.

FIGURE 39 BGP path hiding



Advantages of BGP additional-paths

- Fast convergence and fault tolerance: When BGP additional-paths is enabled, more than one path to a destination is advertised. If one of the paths goes down, connectivity is easily restored due to the availability of backup paths. If the next hop for the prefix becomes unreachable, the device can switch to the backup route immediately without having to wait for BGP control plane messages.

- Enhanced load balancing capabilities: Traditionally with RRs in an iBGP domain, only the best path is given to the clients, even if ECMP paths exist. This affects load balancing. With additional paths advertised by RRs, the clients have more effective load balancing.

Considerations and limitations for BGP additional-paths RIB-in

- When BGP additional-paths is not configured, only one NLRI per prefix per peer is supported. Any additional NLRI update for the same prefix from the same peer replaces the existing one.
- When BGP additional-paths is configured, a device can receive multiple NLRI advertisements for the same prefix from the same peer that are uniquely identified by NLRI path identifiers.
- For Brocade MLX Series and Brocade NetIron XMR Series devices, the maximum number of additional paths per peer per prefix is 128.
- For Brocade NetIron CER Series and Brocade NetIron CES Series devices, the maximum number of additional paths per peer per prefix is 64.
- Changes in the capability of sending or receiving additional paths are reflected only after the BGP session is restarted.

Considerations and limitations for BGP additional-paths RIB-out

- Changes in the capability of sending or receiving additional paths are reflected only after the BGP session is restarted.
- The maximum number of paths that can be advertised per prefix is 16. If there are more than 16 paths for a prefix in the RIB-in, only 16 can be advertised.
- You should maintain the number of RIB-in paths for any prefix in the range of 16 for smooth RIB-out processing. Otherwise the RIB-out processing time increases exponentially in the scaled scenarios.

Upgrade and downgrade considerations

If BGP additional-paths is enabled and the configuration saved, an error message occurs if the software is downgraded to an earlier version. BGP additional-paths should be unconfigured before a downgrade take place.

BGP additional-paths functionality

BGP additional-paths is implemented by including an additional four-octet value known as a path identifier (ID) for each path in the NLRI. Path IDs are unique to a peering session and are generated for each network. A generated Path ID is unique per peer per prefix. A BGP device can receive the same path ID for the same prefix from two different peers, or it can receive the same path ID from the same peer for two different prefixes.

A path ID can apply to the IPv4 or IPv6 unicast or multicast address family.

Therefore, when the same prefix is received with the same path ID from the same peer, it is considered as a replacement route or a duplicate route. When the same prefix is received with a different path ID from the same peer, it is considered as an additional path to the prefix.

To send or receive additional paths, the additional-paths capability must be negotiated. If it is not negotiated, only the best path can be sent. BGP updates carry the path ID once the additional-paths capability is negotiated. In order to carry the path ID in an update message, the existing NLRI encodings are extended by prepending the path ID field, which consists of four octets.

The assignment of the path ID for a path by a BGP device occurs in such a way that the BGP device is able to use the prefix and path ID to uniquely identify a path advertised to a neighbor so as to continue to send further updates for that path. The receiving BGP neighbor that re-advertises a route regenerates its own path ID to be associated with the re-advertised route.

The set of additional paths advertised to each neighbor can be different, and advertisement filters are provided on a per-neighbor basis.

NOTE
BGP additional-paths is supported for the BGP IPv4 and IPv6 unicast address families and the BGP IPv4 and IPv6 multicast address families.

NOTE
BGP additional-paths is not supported for the BGP L2VPN VPLS, BGP VPNv4 unicast, and BGP VPNv6 address families.

There are three basic steps involved in configuring BGP additional-path:

- **Capability Negotiation**: Specify whether the device can send, receive, or send and receive additional paths. This is done at the address family level or peer-group level or the neighbor level. Refer to the sections and the NetIron Command Reference for more information.
- **Select Candidate paths**: Select a set or sets of candidate paths for advertisement by specifying selection criteria. This is done at the address family level.
- **Advertise additional paths from the candidate set**: Advertise to a neighbor a set or sets of additional paths from the candidate paths marked. This is done at the neighbor level or peer-group level.

Configuring BGP4 additional-paths and additional-path selection for the default VRF

You can enable BGP additional-paths send and receive capability for the IPv4 address family and select a set or sets of candidate paths for advertisement by specifying the selection criteria. This task specifies that all BGP paths are eligible to be selected as additional paths under the IPv4 unicast address family for the default VRF and enables BGP additional-paths.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp)# local-as 65520
```

4. Enter the **additional-paths** command, using the **receive** parameter, to enable additional-paths receive capability under the IPv4 unicast address family.

```
device(config-bgp)# additional-paths receive
```

5. Enter the **additional-paths** command, using the **send** parameter, to enable additional-paths send capability under the IPv4 unicast address family.

```
device(config-bgp)# additional-paths send
```

6. Enter the **additional-paths select** command, using the **all** parameter, to specify that all BGP paths are eligible to be selected as additional paths under the IPv4 unicast address family.

```
device(config-bgp)# additional-paths select all
```

7. Enter the **neighbor additional-paths advertise** command, specifying an IP address and using the **all** parameter, to configure BGP to advertise all BGP additional paths to a neighbor.

```
device(config-bgp)# neighbor 10.11.12.13 additional-paths advertise all
```

The following example enables BGP additional-paths send and receive capability and specifies that all BGP paths are eligible to be selected as additional paths under the IPv4 unicast address family. The additional-paths feature is enabled and all paths can be advertised to a BGP neighbor.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# additional-paths receive
device(config-bgp)# additional-paths send
device(config-bgp)# additional-paths select all
device(config-bgp)# neighbor 10.11.12.13 additional-paths advertise all
```

Configuring BGP4 additional-paths and additional-path selection for a non-default VRF instance

You can enable BGP additional-paths send and receive capability under the configured IPv4 address family for a non-default VRF instance and select a set or sets of candidate paths for advertisement by specifying the selection criteria. This task specifies that all BGP paths are eligible to be selected as additional paths under the IPv4 unicast address family for VRF green and enables BGP additional-paths.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp)# local-as 65520
```

4. Enter the **address-family unicast ipv4** command, using the **vrf** parameter, and specifying a VRF, to enter BGP address-family IPv4 unicast VRF configuration mode.

```
device(config-bgp)# address-family ipv4 unicast vrf green
```

5. Enter the **additional-paths** command, using the **receive** parameter, to enable additional-paths receive capability under the configured IPv4 address family for VRF instance green.

```
device(config-bgp)# additional-paths receive
```

6. Enter the **additional-paths** command, using the **send** parameter, to enable additional-paths send capability under the configured IPv4 address family for VRF instance green.

```
device(config-bgp)# additional-paths send
```

7. Enter the **additional-paths select** command, using the **all** parameter, to specify that all BGP paths are eligible to be selected as additional paths under the IPv4 unicast address family for VRF instance green.

```
device(config-bgp-ipv4u-vrf)# additional-paths select all
```

8. Enter the **neighbor additional-paths advertise** command, specifying an IP address and using the **all** parameter, to configure BGP to advertise all BGP additional paths to a neighbor for VRF instance green.

```
device(config-bgp)# neighbor 10.11.12.13 additional-paths advertise all
```

The following example enables BGP additional-paths send and receive capability and specifies that all BGP paths are eligible to be selected as additional paths under the IPv4 unicast address family for VRF instance green. The additional-paths feature is enabled and all paths can be advertised to a BGP neighbor.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# address-family ipv4 unicast vrf green
device(config-bgp-ipv4u-vrf)# additional-paths receive
device(config-bgp-ipv4u-vrf)# additional-paths send
device(config-bgp-ipv4u-vrf)# additional-paths select all
device(config-bgp-ipv4u-vrf)# neighbor 10.11.12.13 additional-paths advertise all
```

Configuring BGP4 additional-paths for a specified neighbor

You can apply filters for the advertisement of additional paths for specified BGP neighbors.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp)# local-as 65520
```

4. Enter the **address-family ipv4 unicast** command to enter BGP address-family IPv4 multicast configuration mode.

```
device(config-bgp)# address-family ipv4 multicast
```

5. Enter the **neighbor additional-paths** command, specifying an IP address and using the **receive** parameter, to enable additional-paths receive capability from a specified BGP neighbor.

```
device(config-bgp-ipv4m)# neighbor 10.11.12.13 additional-paths receive
```

6. Enter the **neighbor additional-paths** command, specifying an IP address and using the **send** parameter, to enable additional-paths send capability to a specified BGP neighbor.

```
device(config-bgp-ipv4m)# neighbor 10.11.12.13 additional-paths send
```

7. Enter the **additional-paths select** command, using the **all** parameter, to specify that all BGP paths are eligible to be selected as additional paths under the configured IPv4 address family.

```
device(config-bgp-ipv4m)# additional-paths select all
```

8. Enter the **neighbor additional-paths advertise** command, specifying an IP address and using the **all** parameter, to configure BGP to advertise all BGP additional paths.

```
device(config-bgp-ipv4m)# neighbor 10.11.12.13 additional-paths advertise all
```

The following example specifies that all BGP paths are eligible to be selected as additional paths under the IPv4 multicast address family and enables the capability to send and receive additional paths for a specified neighbor. The additional-paths feature is enabled and all paths can be advertised to the BGP neighbor.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# address-family ipv4 multicast
device(config-bgp-ipv4m)# neighbor 10.11.12.13 additional-paths receive
device(config-bgp-ipv4m)# neighbor 10.11.12.13 additional-paths send
device(config-bgp-ipv4m)# additional-paths select all
device(config-bgp-ipv4m)# neighbor 10.11.12.13 additional-paths advertise all
```

Configuring BGP additional-paths for a specified BGP4 neighbor for a non-default VRF instance

You can enable the advertisement of additional paths for specified BGP4 neighbors and apply filters for the advertisement of additional paths for BGP neighbors for a non-default VRF instance.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp)# local-as 65520
   ```

4. Enter the **address-family unicast ipv4** command, using the **vrf** parameter and specifying a VRF, to enter BGP address-family IPv4 unicast VRF configuration mode.

   ```
   device(config-bgp)# address-family ipv4 unicast vrf green
   ```

5. Enter the **neighbor additional-paths** command, specifying an IP address and using the **receive** parameter, to enable additional-paths receive capability from a specified BGP neighbor for VRF green.

   ```
   device(config-bgp-ipv4u-vrf)# neighbor 10.11.12.13 additional-paths receive
   ```

6. Enter the **neighbor additional-paths** command, specifying an IP address and using the **send** parameter, to enable additional-paths send capability to a specified BGP neighbor for VRF green.

   ```
   device(config-bgp-ipv4u-vrf)# neighbor 10.11.12.13 additional-paths send
   ```

7. Enter the **additional-paths select** command, using the **all** parameter, to specify that all BGP paths are eligible to be selected as additional paths under the IPv4 address unicast family for VRF green.

   ```
   device(config-bgp-ipv4u-vrf)# additional-paths select all
   ```

8. Enter the **neighbor additional-paths advertise** command, specifying an IP address and using the **all** parameter, to configure BGP to advertise all BGP additional paths for VRF green.

   ```
   device(config-bgp-ipv4u-vrf)# neighbor 10.11.12.13 additional-paths advertise all
   ```

The following example specifies that all BGP paths are eligible to be selected as additional paths under the IPv4 unicast address family for VRF green, and enables BGP additional-paths send and receive capability for a specified neighbor. The additional-paths feature is enabled and all paths can be advertised to the BGP neighbor.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# address-family ipv4 unicast vrf green
device(config-bgp-ipv4u-vrf)# neighbor 10.11.12.13 additional-paths receive
device(config-bgp-ipv4u-vrf)# neighbor 10.11.12.13 additional-paths send
device(config-bgp-ipv4u-vrf)# additional-paths select all
device(config-bgp-ipv4u-vrf)# neighbor 10.11.12.13 additional-paths advertise all
```

Disabling BGP additional-paths for a specified BGP4 neighbor

You can disable the BGP additional-paths capability for a specific BGP neighbor after the capability has been enabled at the peer-group or address-family level.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp)# local-as 65520
   ```

4. Enter the **neighbor additional-paths disable** command, specifying an IP address, to disable the sending of additional paths by BGP4 to the specified BGP neighbor.

   ```
   device(config-bgp)# neighbor 10.11.12.13 additional-paths disable
   ```

The following example disables the sending of additional paths by BGP4 to the specified neighbor in address-family IPv4 unicast configuration mode.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# neighbor 10.11.12.13 additional-paths disable
```

Configuring BGP additional-paths for a BGP peer group

You can enable the advertisement of additional paths for specified BGP peer groups and apply filters for the advertisement of additional paths for BGP peer groups.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp)# local-as 65520
   ```

4. Enter the **neighbor** *peer-group-name* **peer-group** command to create a peer group.

```
device(config-bgp)# neighbor mypeergroup1 peer-group
```

5. Enter the **neighbor** *peer-group-name* **remote-as** command to specify the ASN of the peer group.

```
device(config-bgp)# neighbor mypeergroup1 remote-as 11
```

6. Enter the **neighbor** *ip-address* **peer-group** command to associate a neighbor with the peer group.

```
device(config-bgp)# neighbor 10.11.12.13 peer-group mypeergroup1
```

7. Enter the **neighbor** *ip-address* **peer-group** command to associate a neighbor with the peer group.

```
device(config-bgp)# neighbor 10.11.13.15 peer-group mypeergroup1
```

8. Enter the **neighbor additional-paths** command, specifying a peer group name and using the **receive** parameter, to enable additional-paths receive capability from a specified BGP peer group.

```
device(config-bgp)# neighbor mypeergroup1 additional-paths receive
```

9. Enter the **neighbor additional-paths** command, specifying a peer group name and using the **send** parameter, to enable additional-paths send capability to a specified BGP peer group.

```
device(config-bgp)# neighbor mypeergroup1 additional-paths send
```

10. Enter the **additional-paths select** command, using the **all** parameter, to specify that all BGP paths are eligible to be selected as additional paths.

```
device(config-bgp)# additional-paths select all
```

11. Enter the **neighbor additional-paths advertise** command, specifying a peer group name and using the **all** parameter, to configure BGP to advertise all BGP additional paths.

```
device(config-bgp)# neighbor mypeergroup1 additional-paths advertise all
```

The following example creates a peer group, enables BGP4 additional-paths send and receive capability for the specified BGP peer group, and specifies that all BGP paths are eligible to be selected as additional paths. The additional-paths feature is enabled and all paths can be advertised.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# neighbor mypeergroup1 peer-group
device(config-bgp)# neighbor mypeergroup1 remote-as 11
device(config-bgp)# neighbor 10.11.12.13 peer-group mypeergroup1
device(config-bgp)# neighbor 10.11.13.15 peer-group mypeergroup1
device(config-bgp)# neighbor mypeergroup1 additional-paths receive
device(config-bgp)# neighbor mypeergroup1 additional-paths send
device(config-bgp)# additional-paths select all
device(config-bgp)# neighbor mypeergroup1 additional-paths advertise all
```

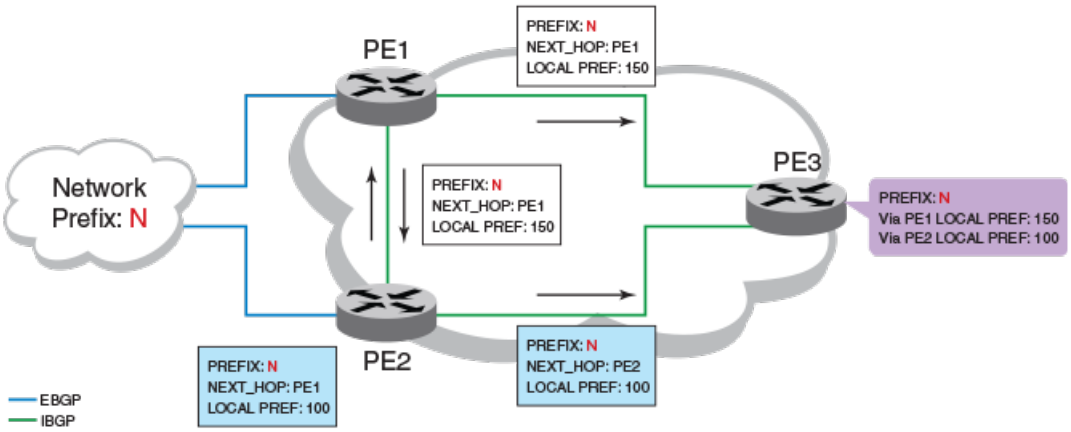
BGP best external overview

BGP best external enables a device to advertise the most preferred route among those received from external neighbors as a backup route.

In active-backup topologies, service providers use routing policies that cause a border router to choose a path received over an Interior Border Gateway Protocol (iBGP) session as the best path for a prefix. This path is chosen even if an Exterior Border Gateway Protocol (eBGP) learned path exists. BGP best external is beneficial in such a topology. In such a topology, one exit or egress point for the prefix in the autonomous system is defined, and the other points are used as backups if the primary link or eBGP peering is unavailable. The border router does not advertise any path for such prefixes, and the paths learned over its eBGP sessions from the autonomous system (AS) are hidden. To cope with this situation, a device can advertise the best external path.

In the following figure, PE1 is the primary path to network N, and PE2 is the backup path. If BGP best external is not configured, PE2 does not advertise prefix N to its iBGP peers because PE2 prefers the iBGP route from PE1 as the best route compared to its best eBGP route. However, if BGP best external is configured, PE2 propagates its best external path to its iBGP peers so that PE3 has two paths for prefix N.

FIGURE 40 BGP best external



NOTE
BGP best external is supported for the BGP IPv4 and IPv6 unicast address families and the BGP IPv4 and IPv6 multicast address families.

NOTE
BGP best external is not supported for the BGP L2VPN VPLS, BGP VPNv4 unicast, and BGP VPNv6 address families.

Limitations of BGP best external

- BGP best external advertises the best external path to iBGP peers only.
- When BGP best external is configured on a route reflector (RR), the best external path is not advertised.

Upgrade and downgrade considerations

If BGP best external is enabled and the configuration saved, an error message occurs if the software is downgraded to an earlier version. BGP best external should be unconfigured before a downgrade takes place.

Configuring BGP4 best external

You can enable BGP4 to calculate the best external path and to advertise this path to its neighbors.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp)# local-as 65520
   ```

4. Enter the **advertise-best-external** command to configure BGP4 to calculate the best external path and to advertise this path to its neighbors.

   ```
   device(config-bgp)# advertise-best-external
   ```

The following example configures BGP4 to calculate the best external path and to advertise this path to its neighbors under the IPv4 unicast address family .

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# advertise-best-external
```

BGP additional-paths

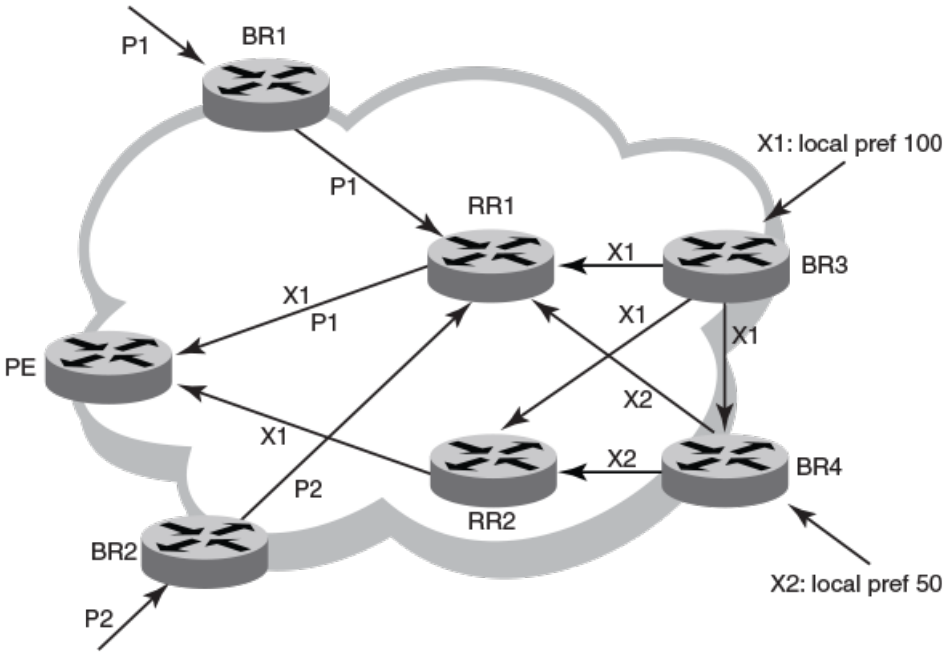
BGP additional-paths enables the advertisement of multiple paths for the same prefix without new paths implicitly replacing the previous paths. Path diversity is achieved rather than path hiding.

A BGP device generally advertises only its best path to neighboring devices, even when multiple paths exist; the advertisement of the same prefix from the same neighbor replaces the previous announcement of that prefix. This path hiding is an implicit withdrawal behavior, which achieves better scaling but at the cost of path diversity. Path hiding can affect the efficient use of BGP multipath and path diversity, and prevent hitless planned maintenance. Upon next-hop failures, path hiding also inhibits fast and local recovery because the network must wait for BGP control plane convergence to restore traffic.

The following figure shows two types of path hiding:

- Paths P1 and P2 (for prefix P) are advertised to RR1; P1 is advertised from BR1 and P2 is advertised from BR2. RR1 selects P1 as the best path and advertises only P1 to PE.
- Paths X1 and X2 (for prefix X) are advertised to BR4; X1 is advertised from BR3 with a local preference of 100. BR4 also has path X2 with a local preference 50. However, only the best path (X1) is selected. BR3 advertises X1 to the RRs and X2 is suppressed.

FIGURE 41 BGP path hiding



BGP additional-paths enables BGP to advertise even secondary best routes, so that multiple paths for the same prefix can be advertised without the new paths implicitly replacing previous paths. BGP additional-paths provides a generic way of offering path diversity.

BGP additional-paths is implemented by including an additional four-octet value known as a path identifier (ID) for each path in the Network Layer Reachability Information (NLRI). Path IDs are unique to a peering session and are generated for each network; that is, a generated path ID is unique for each peer for each prefix.

A BGP device can receive the same path ID for the same prefix from two different peers, or it can receive the same path ID from the same peer for two different prefixes:

- When the same prefix is received with the same path ID from the same peer, it is considered to be a replacement route or duplicate route.
- When the same prefix is received with a different path ID from the same peer, it is considered to be an additional path to the prefix.

To send or receive additional paths, the additional-paths capability must be negotiated. When it is not negotiated, only the best path is sent.

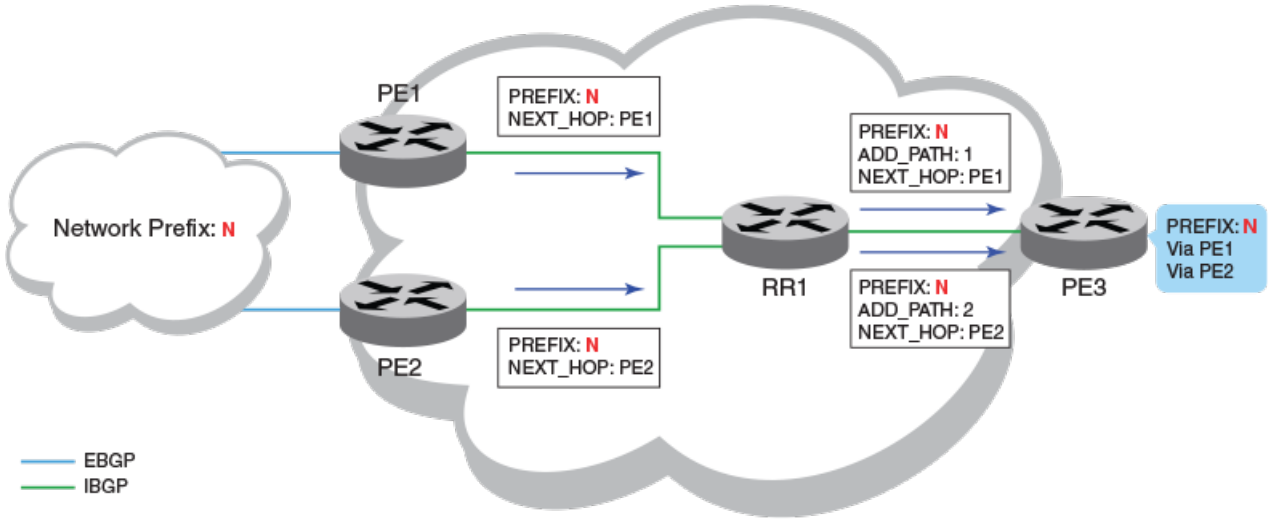
When the additional-paths capability is negotiated, BGP updates carry the path ID. To carry the path ID in an update message, the existing NLRI encodings are extended by prepending the path ID field, which consists of four octets.

The assignment of the path ID for a path by a BGP device occurs in such a way that the BGP device is able to use the prefix and path ID to uniquely identify a path advertised to a neighbor so as to continue to send further updates for that path. The receiving BGP neighbor that readvertises a route regenerates its own path ID that is to be associated with the readvertised route.

The set of additional paths advertised to each neighbor can be different, and advertisement filters are provided for each neighbor.

The following figure shows a BGP additional-paths configuration that supports path diversity.

FIGURE 42 BGP additional-paths



- Although route reflector RR1 has two sources for prefix N, it advertises only the best route to its peers when the additional-paths capability is not configured.
- When the additional-paths capability is configured at RR1 and PE3, PE3 receives two routes to prefix N. Receiving two routes facilitates quick network convergence and recovery:
  - Using PE1 and PE2
  - ECMP paths to PE1 and PE2

  NOTE
  BGP additional-paths is supported for BGP IPv4 and IPv6 unicast address families.

  NOTE
  BGP additional-paths is not supported for BGP VPNv4 unicast, BGP VPNv6, and BGP EVPN address families.

Configuration of BGP additional-paths

The configuration of BGP additional-paths involves the following actions:

- Capability negotiation—Specifying, at the address family, peer group, or neighbor levels, whether the device can send, receive, or both send and receive additional-paths .
- Candidate path selection—Specifying, at the address family level, selection criteria for a set or sets of candidate paths to be advertised.
- Advertisement of additional-paths from the candidate set—Specifying, at the neighbor or peer group levels, a set or sets of additional-paths to advertise to a neighbor. The additional-paths to advertise must be from the candidate paths that are marked.

Advantages of BGP additional-paths

BGP additional-paths supports fast convergence and fault tolerance and enhances load-balancing capabilities.

- Fast convergence and fault tolerance: When BGP additional-paths is enabled, more than one path to a destination is advertised. When one of the paths goes down, connectivity is easily restored due to the availability of backup paths. When the next-hop for

the prefix becomes unreachable, the device can switch to the backup route immediately without having to wait for BGP control plane messages.

- Enhanced load-balancing capabilities: Traditionally with route reflectors (RRs) in an iBGP domain, only the best path is given to clients, even when ECMP paths exist. Giving only the best path affects load balancing. When additional paths are advertised by RRs, the clients have more effective load balancing.

Considerations and limitations for BGP additional-paths

The following considerations and limitations apply to BGP additional-paths.

- BGP additional-paths is supported for the following BGP address families:
  - IPv4 unicast
  - IPv4 unicast VRF
  - IPv6 unicast
  - IPv6 unicast VRF
- Changes in the capability of receiving or sending additional-paths are reflected only after the BGP session is restarted.
- RIB-in
  - When BGP additional-paths is not configured, only one NLRI for each prefix for each peer is supported. Any additional NLRI update for the same prefix from the same peer replaces the existing one.
  - When BGP additional-paths is configured, a device can receive multiple NLRI advertisements for the same prefix from the same peer that are uniquely identified by NLRI path identifiers.
  - The maximum number of additional-paths for each peer for each prefix is 128.
  - For Brocade MLX Series and Brocade NetIron XMR Series devices, the maximum number of additional paths per peer per prefix is 128.
  - For Brocade NetIron CER Series and Brocade NetIron CES Series devices, the maximum number of additional paths per peer per prefix is 64.
- RIB-out
  - The maximum number of paths that can be advertised for each prefix is 16. When more than 16 paths for a prefix exist in the RIB-in, only 16 can be advertised.
  - For smooth RIB-out processing, the number of RIB-in paths for any prefix should be maintained within the range from 1 through 16: RIB-out processing time increases exponentially in the scaled scenarios.

Upgrade and downgrade considerations for BGP additional-paths

BGP additional-paths configuration should be removed before a downgrade to a version of software that does not support additional-paths.

When BGP additional-paths is enabled and the configuration saved, an error message occurs when the software is downgraded to an earlier version that does not support the feature. To avoid this error message, the BGP additional-paths configuration should be removed before the downgrade takes place and reconfigured when upgraded again.

Configuring BGP additional-paths and additional-path selection at address-family level

BGP additional-paths is configured by enabling the additional-paths capability and specifying paths that are candidates for selection as additional-paths.

BGP additional-paths is supported for the following BGP address families:

- IPv4 unicast
- IPv4 unicast VRF
- IPv6 unicast
- IPv6 unicast VRF

Perform the following task to enable the additional-paths capability and specify paths that are candidates for selection as additional-paths, under the IPv4 unicast address family.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter configuration mode for the IPv4 unicast address family.

   ```
   device(config-bgp)# address-family ipv4 unicast
   ```

4. Enable receiving and sending of additional paths.

   ```
   device(config-bgp-ipv4u)# additional-paths receive send
   ```

   A device can be configured to receive, send, or receive and send additional paths.

   NOTE
   A change to the additional-paths capability is activated only after restarting the BGP session.

5. Specify paths that are candidates for selection as additional-paths. The following example specifies that all paths are candidates for selection as additional-paths.

   ```
   device(config-bgp-ipv4u)# additional-paths select all
   ```

   The **all** option configures all (up to a maximum of 16) paths as candidates for selection as additional-paths.

The following example enables the receiving and sending of BGP additional-paths and specifies that all BGP paths are eligible for selection as additional-paths under the IPv4 unicast address family.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# additional-paths receive send
device(config-bgp-ipv4u)# additional-paths select all
```

After completing this task, restart the BGP session to activate the new additional-paths capability configuration.

Configuring BGP additional-paths and additional-path advertisement at neighbor level

BGP additional-paths can be enabled for a specific peer device (neighbor) under a BGP address family. In addition, configuration options are available to control the additional-paths that are advertised to neighbors.

Before completing this task, the set of paths eligible for selection as additional-paths must be configured by using the **additional-paths select** command, under the relevant address family. An example is provided at the end of this task that shows all the configuration steps in order.

BGP additional-paths is supported for the following BGP address families:

- IPv4 unicast
- IPv4 unicast VRF
- IPv6 unicast
- IPv6 unicast VRF

Perform the following task to enable additional-paths on a specific peer device and to specify the additional routes to be advertised by the peer, under the IPv4 unicast address family.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter IPv4 unicast address family configuration mode.

   ```
   device(config-bgp)# address-family ipv4 unicast
   ```

4. Enable additional-paths for a specific peer.

   ```
   device(config-bgp-ipv4u)# neighbor 10.60.60.20 additional-paths receive send
   ```

   This example enables the capability to both receive and send additional-paths on peer device 10.60.60.20.

   > **NOTE**
   > Changes in the capability of receiving or sending additional-paths are reflected only after the BGP session is restarted.

5. Configure the additional-paths to be advertised by the peer.

   ```
   device(config-bgp-ipv4u)# neighbor 10.60.60.20 additional-paths advertise all
   ```

   Paths configured for advertisement to neighbors must be a subset of the paths previously configured by using the **additional-paths select** command under the IPv4 address family.

6. Verify the configuration.

```
device# show ip bgp neighbors

    '+': Data in InQueue '>': Data in OutQueue '-': Clearing
    '*': Update Policy 'c': Group change 'p': Group change Pending
    'r': Restarting 's': Stale '^': Up before Restart '<': EOR waiting

1   IP Address: 10.60.60.20, AS: 200 (IBGP), RouterID: 10.60.60.20, VRF: default-vrf
    State: ESTABLISHED, Time: 4h3m28s, KeepAliveTime: 60, HoldTime: 180
       KeepAliveTimer Expire in 0 seconds, HoldTimer Expire in 159 seconds
    Minimal Route Advertisement Interval: 0 seconds
       RefreshCapability: Received
    Address Family : IPV4 Unicast
       Configured with Add-Path(send receive)capability
       Received Add-Path (send receive)capability in open msg
       Negotiated Add-Path(send receive)capability
    Messages:    Open        Update      KeepAlive   Notification   Refresh-Req
       Sent    : 1           1           275         0              0
       Received: 1           1           275         0              0
    Last Update Time: NLRI       Withdraw          NLRI       Withdraw
                  Tx: 4h3m28s    ---           Rx: 4h3m28s    ---
```

The following example first shows how to configure all paths as eligible for selection as additional-paths under the IPv4 address family. It then shows how to enable receiving and sending additional-paths on a specific peer device (10.60.60.20) and configures the advertisement of all paths by 10.60.60.20.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# additional-paths select all

device(config-bgp-ipv4u)# neighbor 10.60.60.20 additional-paths receive send
device(config-bgp-ipv4u)# neighbor 10.60.60.20 additional-paths advertise all
```

After completing this task, restart the BGP session to activate the new additional-paths capability configuration.

Configuring additional-paths and advertisement of additional-paths at peer group level

BGP additional-paths can be enabled for a specific peer group (neighbor) under a BGP address family. In addition, configuration options are available to control the additional-paths that are advertised by the neighbor.

Before completing this task, configure (by using the **additional-paths select** command) the set of paths eligible for selection as additional-paths under the relevant address family. An example is provided at the end of this task that shows all the configuration steps in order.

BGP additional-paths is supported for the following BGP address families:

- IPv4 unicast
- IPv4 unicast VRF
- IPv6 unicast
- IPv6 unicast VRF

Perform the following task to enable additional-paths for a specific peer group and specify the additional routes for the peer group to advertise, under the IPv4 unicast address family.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing.

```
device(config)# router bgp
```

3. Create a peer group. The following example shows how to create a peer group named pg-1

```
device(config-bgp)# neighbor pg-1 peer-group
```

4. Enter IPv4 unicast address family configuration mode.

```
device(config-bgp)# address-family ipv4 unicast
```

5. Enable additional-paths for the peer group.

```
device(config-bgp-ipv4u)# neighbor pg-1 additional-paths receive send
```

This example enables the capability to both receive and send additional-paths on peer group pg-1.

> **NOTE**
> Changes in the capability of receiving or sending additional-paths are reflected only after the BGP session is restarted.

6. Configure the additional-paths to be advertised by the peer group. The following example shows how to configure the advertisement of group-best paths.

```
device(config-bgp-ipv4u)# neighbor pg-1 additional-paths advertise group-best
```

Paths configured for advertisement to neighbors must be a subset of the paths previously configured by using the **additional-paths select** command under the IPv4 address family.

The following example first shows how to configure **group-best** paths as candidates for selection as additional paths, under the IPv4 address family. It then shows how to enable receiving and sending additional-paths for peer group pg-1 and configures the advertisement of group-best paths for pg-1.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# additional-paths select group-best
device(config-bgp-ipv4u)# exit

device(config-bgp)# neighbor pg-1 peer-group
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor pg-1 additional-paths receive send
device(config-bgp-ipv4u)# neighbor pg-1 additional-paths advertise group-best
```

After completing this task, restart the BGP session to activate the new additional-paths capability configuration.

Filtering additional-paths advertised at route-map level

You can configure a route map instance to filter the additional-paths that are advertised.

Before completing this task, the set of paths eligible for selection as additional-paths must be configured by using the **additional-paths select** command, under the relevant address family. An example is provided at the end of this task that shows all the configuration steps in order.

Perform the following task to configure a route-map instance to filter the additional-paths advertised.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Create a route map instance and enter configuration mode for the route map instance.

```
device(config)# route-map rm-1 permit 123
```

   This example creates instance 123 of a route map named rm-1 and enters configuration mode for the instance.

3. Configure filtering of additional-paths advertised for the route-map instance.

```
device(config-route-map/permit/123)# match additional-paths advertise-set best-range 2 13
```

   This example configures the route-map instance to advertise paths with a path marking (tag) in the range from 2 through 13.

   NOTE
   Only one **match additional-paths advertise-set** configuration is allowed for a route map instance; any subsequent **additional-paths advertise-set** configuration overwrites the previous configuration.

   NOTE
   You cannot use a route map that is configured with an additional-paths match statement for incoming routes (routes that are configured by using the **neighbor route-map** command specifying the **in** option.

The following example first shows how to configure all paths to be eligible for selection as additional-paths, under the IPv4 address family. It then configures a route map (rm-1) to advertise paths marked in the range from 2 through 14.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# additional-paths select all
device(config-bgp-ipv4u)# end

device(config)# route-map rm-1 permit 123
device(config-route-map/permit/123)# match additional-paths advertise-set best-range 2 13
```

Disabling BGP additional-paths capability for a neighbor

The additional-paths capability can be disabled for a specific neighbor or peer group under an address family.

When an additional-paths capability is configured for a BGP address family, you can disable the capability for a specific neighbor or peer group under the address family.

Perform the following task to disable the additional-paths capability for a specific neighbor.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing.

```
device(config)# router bgp
```

3. Enter configuration mode for the IPv4 unicast address family.

```
device(config-bgp)# address-family ipv4 unicast
```

4. Disable the additional-paths capability for the specific peer device 10.12.12.12.

```
device(config-bgp-ipv4u)# neighbor 10.12.12.12 additional-paths disable
```

   When an additional-paths capability is not previously configured at address-family level and this command is issued, an error message is displayed.

The following example shows how to disable the additional-paths capability on the specific neighbor 10.12.12.12 within the IPv4 unicast address family.

```
evice# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.12.12.12 additional-paths disable
```

The following example shows how to re-enable the additional-paths capability on the specific neighbor 10.12.12.12 within the IPv4 unicast address family.

```
evice# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# no neighbor 10.12.12.12 additional-paths disable
```

After completing this task, restart the BGP session to activate the new additional-paths capability configuration.

BGP best-external route

The best external route is the most preferred route among those received from external neighbors. Advertising the best external route supports fast restoration of connectivity.

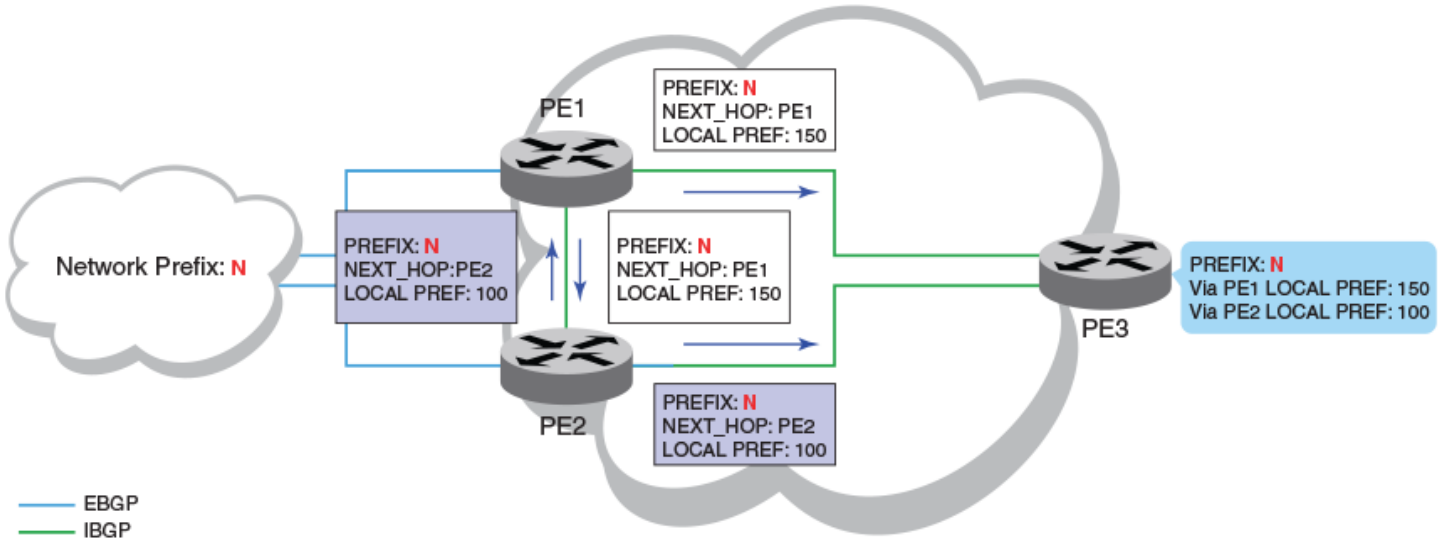
The best external route, when different from the best route, acts as a backup route and introduces additional information into an Interior Border Gateway Protocol (iBGP) mesh, which may be useful in restoration of connectivity.

In active-backup topologies, service providers may use routing policies that cause a border router to choose a path received over an iBGP session (of another border router) as the best path for a prefix even when an Exterior Border Gateway Protocol (eBGP) learned path exists on the device. This type of active-backup topology defines one exit or egress point for the prefix in the autonomous system and uses the other points as backups when the primary link or eBGP peering is not available. When advertisement of the best-external path is not configured, such a routing policy causes the border router to hide (from the autonomous system) the paths learned over its eBGP sessions because it does not advertise these paths.

When advertisement of the best external path is configured by using the **best-external** command, the best external learned path, when different from the best route, is advertised and acts as a backup route that supports fast restoration of connectivity.

The following figure illustrates the impact of best-external configuration in a BGP topology.

FIGURE 43 BGP best-external configuration



- PE1 is the primary path and PE2 is the backup path to network N
- When advertisement of the best external path is not configured:
  - PE2 does not advertise prefix N to its iBGP peers because PE2 prefers the iBGP route from PE1, which has a higher local preference, as the best route when compared to its best external (eBGP) route
  - PE3 has one path for prefix N
- When advertisement of the best external path is configured:
  - PE2 propagates its best external path to its iBGP peers
  - PE3 has two paths for prefix N

Configuring BGP best-external route at address-family level

When the best external route is advertised to internal peers, it acts as a backup route that may be useful in the restoration of connectivity.

Perform the following task to store the best external route (in addition to the best route) and advertise it to internal peers under the IPv4 unicast address family.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing.

```
device(config)# router bgp
```

3. Enter IPv4 unicast address family command mode.

```
device(config-bgp)# address-family ipv4 unicast
```

4. Configure the storage and advertisement, to internal peers, of the best external route.

```
device(config-bgp-ipv4u)# advertise-best-external
```

5. Return to privileged EXEC mode.

```
device(config-bgp-ipv4u)# end
```

6. Verify the configuration.

```
device# show ip bgp

Total number of BGP Routes: 4
Status codes: s suppressed, d damped, h history, * valid, > best, i internal, S stale, x best-
external
Origin codes: i - IGP, e - EGP, ? - incomplete
    Network           Next Hop        MED        LocPrf     Weight Path
*>i 110.110.110.0/24   50.50.50.10                150        0      i
*x  110.110.110.0/24   20.20.20.10                100        0      200 i
*   110.110.110.0/24   30.30.30.10                100        0      300 i
*   110.110.110.0/24   40.40.40.10                100        0      400 i
```

The following example shows how to store the best external route and advertise it to internal peers, under the IPv4 unicast address family.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# advertise-best-external
```

Auto shutdown of BGP neighbors on initial configuration

The auto shutdown of BGP neighbors on initial configuration feature prevents a BGP peer from attempting to establish connections with remote peers when the BGP peer is first configured. Sessions are only initiated when the entire configuration for the BGP peer is complete.

When the auto shutdown of BGP neighbors on initial configuration feature is enabled, the value of the shutdown parameter for any existing configured neighbor is not changed. Any new BGP neighbor configured after the feature is enabled has the shutdown state set to the configured value. When the feature is enabled and a new BGP neighbor is configured, the shutdown parameter of the BGP neighbor is automatically set to enabled. When all the BGP neighbor parameters are configured and it is ready to start the establishment of BGP session with the remote peer, the shutdown parameter of the BGP neighbor is then disabled.

Any new neighbor configured and added to a peer group has the shutdown flag enabled by default. Additionally, any new neighbor configured with the autonomous system (AS) in which that remote neighbor resides specified using the **neighbor remote-as** command has the shutdown flag enabled by default.

Configuring auto shutdown of BGP neighbors on initial configuration

Follow this procedure to enable auto shutdown of BGP neighbors on initial configuration.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp)# local-as 65520
```

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **auto-shutdown-new-neighbors** command to prevent the BGP peer from attempting to establish connections with remote peers until all BGP neighbor parameters are configured.

```
device(config-bgp)# auto-shutdown-new-neighbors
```

```
device(config-bgp-router)# auto-shutdown-new-neighbors
```

The following example enables auto shutdown of BGP neighbors on initial configuration.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# auto-shutdown-new-neighbors

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# auto-shutdown-new-neighbors

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# auto-shutdown-new-neighbors
```

Disabling the BGP4 peer shutdown state

When the auto shutdown of BGP4 neighbors on initial configuration feature is enabled, a BGP4 peer is prevented from attempting to establish connections with remote peers when the BGP4 peer is first configured. Once all of the configuration parameters for the peer are complete, you can start the BGP4 session establishment process and disable the peer shutdown state. Follow this procedure to disable the peer shutdown state.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp)# local-as 65520
```

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **no neighbor shutdown** command, specifying an IP address, to disable the peer shutdown state and begin the BGP4 session establishment process.

```
device(config-bgp)# no neighbor 10.1.1.1 shutdown

device(config-bgp-router)# no neighbor 10.1.1.1 shutdown
```

The following example disables the peer shutdown state and begins the BGP4 session establishment process.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# no neighbor 10.1.1.1 shutdown
```

The following example disables the peer shutdown state and begins the BGP4 session establishment process.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# no neighbor 10.1.1.1 shutdown
```

The following example disables the peer shutdown state and begins the BGP4 session establishment process.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# no neighbor 10.1.1.1 shutdown
```

BGP4 graceful shutdown

BGP4 graceful shutdown automatically reroutes traffic to an alternate path before shutting down a neighbor, minimizing traffic loss during a planned shutdown and re-establishment of BGP4 sessions.

BGP protocol is heavily used in the data center solutions where, for resiliency purposes, redundant spine & leaf devices are deployed to reduce the consequence of any breakdowns. For further information on data center environments, refer to *Extreme SLX-OS IP Fabrics Configuration Guide*.

In the context of a planned maintenance operation, such as firmware upgrade, software upgrade, line card replacement, SFP replacement, cable replacement or node replacement, it may be necessary to shut down one or more BGP peer links or a complete switch. This will result in some traffic disruption until the underlay network converges, even when alternate paths are available.

When an alternate path is available, it is possible to reroute traffic on the backup path before the BGP sessions are torn down (Make before Break) by configuring BGP graceful shutdown. BGP graceful shutdown is an automated procedure that reroutes traffic to an alternate path before shutting down the neighbor, by using BGP attributes like LOCAL_PREFERENCE, AS_PATH PREPEND and GRACEFUL_SHUTDOWN community to make itself less preferred.

The following table describes the default BGP graceful shutdown attributes that are used to make routes less preferable.

NOTE
You can override these parameters by using a route map.

TABLE 25 Default BGP graceful shutdown attributes

Attribute	Value	Description
LOCAL_PREFERENCE	1	The LOCAL_PREFERENCE attribute is used between iBGP & Confederation peer devices.
GRACEFUL_SHUTDOWN community	0xFFFF0000	The GRACEFUL_SHUTDOWN community attribute defined by IANA.

TABLE 25 Default BGP graceful shutdown attributes (continued)

Attribute	Value	Description
AS_PATH PRREPEND	Prepend local AS 3 times	AS_PATH PREPEND is used between eBGP peers.
GRACEFUL_SHUTDOWN timer	600 sec	Time between initiation of the BGP Graceful Shutdown and the actual shutdown of BGP neighbors on the initiator device.

When planned maintenance is carried out, traffic loss is generally expected until BGP converges in the network. By reducing the loss of inbound or outbound traffic flows forwarded along a peering link that is shut down for planned maintenance, BGP graceful shutdown provides benefits, such as fewer lost packets and less time spent reconfiguring devices.

BGP graceful shutdown helps to mitigate traffic loss so that traffic is automatically rerouted before the neighbor is shutdown.

When BGP graceful shutdown is initiated on one or more BGP neighbors, BGP starts a graceful shutdown timer on the graceful shutdown initiator and then performs a route refresh to all graceful shutdown neighbors with updated BGP attributes, such as LOCAL_PREFERENCE for confederations or iBGP, or AS_PATH PREPEND for eBGP. The best path calculation on all graceful shutdown neighbors learned routes is then triggered, making the routes via the graceful shutdown initiator less preferable. Thus, the traffic from graceful shutdown neighbors is re-routed to alternate paths. However, the path via the graceful shutdown initiator is still valid until the graceful shutdown timer expires. Once the graceful shutdown timer expires, all the graceful shutdown neighbors are shut down on the graceful shutdown initiator and all traffic is rerouted to alternate paths. The operator can now carry the maintenance operation.

> NOTE
> You should wait for the timer to expire before carrying out a maintenance operation. If the graceful shutdown procedure is deactivated while the timer is running, the route refresh using the graceful shutdown parameters aborts, a new route refresh is triggered, and the graceful shutdown timer stops.

For the graceful shutdown initiator, when there is an outbound policy associated with a particular graceful shutdown neighbor, the outbound policy is applied before the graceful shutdown parameters are applied.

For the graceful shutdown neighbor, when there is an inbound policy associated with the graceful shutdown initiator modifying the LOCAL_PREFERENCE or AS_PATH, the GRACEFUL_SHUTDOWN parameters become insignificant. To prevent this, it is recommended that you modify the existing inbound policy rule to match the graceful shutdown community attribute and make the route less preferred. Therefore, the send community attribute should also be negotiated between the graceful shutdown initiator and the neighbor.

BGP4 graceful shutdown supports high availability (HA). During a reboot when graceful shutdown is configured for a peer, the peer is instantly shut down and the graceful shutdown timer is not started.

> NOTE
> The graceful shutdown community attribute should not be considered for aggregation. When an autonomous system (AS) aggregates multiple routes under a cover prefix, to avoid the graceful shutdown behavior being applied to the entire BGP aggregate route, the graceful shutdown community should be filtered out of the aggregate route by using the **no match community** command.

This implementation of BGP4 Graceful Shutdown conforms to RFC 6198 and Graceful BGP session shutdown draft-ietf-grow-bgp-gshut-03.

Limitations

- Before the graceful shutdown community attribute can be sent, the send community must be negotiated.

- Modifications made to a route map during the graceful-shutdown period are not effective for graceful shutdown advertisement routes. The graceful-shutdown period is set by using the **neighbor graceful-shutdown** command specifying the **graceful-shutdown** parameter.

Configuring graceful shutdown for all BGP4 neighbors

Configuring graceful shutdown for all BPG4 neighbors minimizes traffic loss during a planned shutdown of all neighbors by rerouting traffic to alternate paths before the shutdown.

Perform the following steps to configure graceful shutdown for all BGP4 neighbors.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

5. Add a neighbor.

   ```
   device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
   ```

   Repeat this step for all neighbors on the device.

6. Configure graceful shutdown for all neighbors.

   ```
   device(config-bgp-router)# graceful-shutdown 600 community 10
   ```

   This example configures graceful shutdown for all neighbors after 600 seconds and sets the community attribute to 10. The **graceful-shutdown** command gracefully shuts down all neighbors on the node. When a route map is not specified, default graceful shutdown parameters are applied. When a route map is specified, only route-map parameters are applied.

7. Return to privileged EXEC mode.

   ```
   device(config-bgp-router)# end
   ```

8. Verify the configuration.

   ```
   device# show running-configuration router bgp

    router bgp
     local-as 1000
     graceful-shutdown 600 community 10
   !
   ! example truncated for brevity
   ```

The following example shows how to configure graceful shutdown of all BGP4 neighbors after 600 seconds. The community attribute is also set to 10.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# graceful-shutdown 600 community 10
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
!
!
device(config-bgp-router)# graceful-shutdown 600 community 10
```

Configuring graceful shutdown for a BGP4 peer group

Configuring graceful shutdown for a BGP4 peer group minimizes traffic loss during a planned shutdown of the peer group by rerouting traffic to an alternate path before the shutdown.

Perform the following steps to configure graceful shutdown for all peers in a BGP4 peer group.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

5. Add a peer group.

   ```
   device(config-bgp-router)# neighbor grp-1 peer-group
   ```

   This example configures a peer group named grp-1.

6. Configure graceful shutdown for the peer group.

   ```
   device(config-bgp-router)# neighbor grp-1 graceful-shutdown 600 community 20
   ```

   This example configures graceful shutdown of the peer group named grp-1 after 600 seconds and sets the community attribute to 20. This command gracefully shuts down all the peers in a peer-group. When a route map is not specified, default graceful shutdown parameters are applied. When a route map is specified, only route-map parameters are applied.

7. Return to privileged EXEC mode.

   ```
   device(config-bgp-router)# end
   ```

8. Verify the configuration.

```
device# show running-configuration router bgp

 router bgp
  local-as 1000
  neighbor grp-1 peer-group
  neighbor grp-1 graceful-shutdown 600 community 20
!
! example truncated for brevity
```

The following example shows how to configure graceful shutdown of a peer group named grp-1 after 600 seconds. The community attribute is also set to 20.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor grp-1 peer-group
device(config-bgp-router)# neighbor grp-1 graceful-shutdown 600 community 20

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor grp-1 peer-group
device(config-bgp-router)# neighbor grp-1 graceful-shutdown 600 community 20
```

Configuring graceful shutdown for a specific BGP4 neighbor

Configuring graceful shutdown for a specific BGP4 neighbor minimizes traffic loss during a planned shutdown of a single neighbor by rerouting traffic to an alternate path before the shutdown.

Perform the following steps to configure graceful shutdown of a link to a specific BGP4 neighbor.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enable BGP routing.

```
device(config-rbridge-id-122)# router bgp

device(config)# router bgp
```

4. Configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Add a neighbor.

```
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
```

6. Configure graceful shutdown of the neighbor.

```
device(config-bgp-router)# neighbor 10.11.22.13 graceful-shutdown 600 community 30
```

   This example configures graceful shutdown of the neighbor (10.11.22.13) after 600 seconds and sets the community attribute to 30.

7. Return to privileged EXEC mode.

```
device(config-bgp-router)# end
```

8. Verify the configuration.

```
device# show running-configuration router bgp

 router bgp
  local-as 1000
  neighbor 10.11.12.13 remote-as 2
  neighbor 10.11.12.13 graceful-shutdown 600 community 30
!
! example truncated for brevity
```

The following example shows how to configure graceful shutdown of a specific BGP4 neighbor (10.11.22.13) after 600 seconds. The community attribute is also set to 30.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# neighbor 10.11.22.13 graceful-shutdown 600 commuity 30

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# neighbor 10.11.22.13 graceful-shutdown 600 commuity 30
```

BGP automatic neighbor discovery

BGP automatic neighbor discovery is primarily used for automatically bringing BGP up in the underlay network of an IP Fabric, significantly reducing the configuration involved.

When BGP automatic neighbor discovery is configured, individual neighbors and remote AS configurations do not need to be configured. Only the local AS and automatic neighbor discovery need to be configured on one of the BGP peer groups.

In addition, for an IP Fabric, when a new leaf or spine node is added to the topology, only the local AS, the peer group automatic discovery, and password configurations are needed on the newly added node. No further configuration changes are required on the existing nodes of the IP Fabric. Therefore, the situation where the neighbor remote AS needs to be configured on each leaf node for each of the spine nodes and on each spine node, for each of the leaf nodes, is avoided.

> **NOTE**
> For security purposes, a peer group must have a Message Digest 5 (MD5) password. The MD5 password created should be the same across all nodes.

> **NOTE**
> The primary use of BGP automatic neighbor discovery is for automatically bringing BGP up in the underlay network of the IP Fabric topology. For more information on IP Fabrics refer to *Extreme Network OS IP Fabrics Configuration GuideExtreme SLX-OS IP Fabrics Configuration Guide*.

BGP automatic neighbor discovery is achieved by using a new Link Layer Discovery Protocol (LLDP) Type Length Value (TLV) for the BGP information. When BGP automatic neighbor discovery is enabled, each node plays the role of both sender and receiver.

BGP automatic neighbor discovery supports both IPv4 and IPv6 addressing.

The role of sender

- BGP informs LLDP whenever both the local AS is configured and the **neighbor accept-lldp-neighbors** command is set, for one of the peer groups on which an MD5 password is configured. BGP also informs LLDP when the local AS, **neighbor accept-lldp-neighbors** or MD5 password configuration is modified or removed.
- Upon receiving the local AS information from BGP, LLDP broadcasts the information across all the router ports on the device using the new LLDP TLV.
- Upon receiving the local AS information, BGP-identifier, Group-ID, and password from BGP, LLDP broadcasts the information across all the router ports on the device using LLDP sub-TLVs.
- LLDP also encodes the IP address of the respective router port, or the donor IP in the case of un-numbered interfaces in the new TLV.
- LLDP also encodes the IP address (IPv4 or IPv6) of the respective router port, or the donor IP in the case of un-numbered interfaces in the new TLV.
- LLDP maintains this information and periodically advertises it.
- When an interface, which is configured as a router port, comes up or goes down, LLDP updates and sends the TLV accordingly.

The role of receiver

- When LLDP receives the new TLV with the BGP local AS and IP Address information from the peer, it stores the information and forwards it to BGP.
- When LLDP receives the new TLV with the BGP local AS, IP address (IPv4 or IPv6), Group-ID, BGP Identifier, and password information from the peer, it stores the information and forwards it to BGP.
- BGP then creates a peer with the received IP address as the neighbor IP and the received local AS numbers as the neighbor remote AS. Once the peer is created, a TCP session is initiated.
- BGP then creates a peer with the received IP address as the neighbor IP address (IPv4 or IPv6), and the received local AS number as the remote AS of the neighbor. Once the peer is created, a TCP session is initiated.
- The same process is repeated on all the nodes in the IP Fabric, automatically bringing up BGP in the underlay network.

BGP automatic neighbor discovery limitations

The following limitations apply when configuring BGP4 automatic neighbor discovery.

BGP automatic neighbor discovery is:

- Supported only for IPv4.
- Supported only on router ports.
- Supported only for the default VRF.
- Supported for only one peer group identifier.
- Supported for Layer 3 PO and unnumbered interfaces.
- Not supported for BGP multihop sessions.

  However, to support BGP sessions between loopback configurations and on un-numbered interfaces, you can configure **ebgp-multihop** for the peer group on which **accept-lldp-neighbors** is already configured or configure **accept-lldp-neighbors** for the peer group on which **ebgp-multihop** is already configured.
- Not supported for BGP Confederations.
- Not supported for TCP Authentication Option (TCP-AO)

- Not supported on port-channel and virtual Ethernet (VE) interfaces.

In addition:

- The BGP automatic neighbor discovery using LLDP and the dynamic listen range features are mutually exclusive.
- BGP automatic neighbor discovery does not support a second local AS in the BGP local AS sub-TLV.
- BGP4 automatic neighbor discovery works only between supported Extreme devices. Interoperability with other vendors is not possible.

  NOTE
  BGP automatic neighbor discovery is primarily used for automatically bringing BGP up in the underlay network of the IP Fabric Topology. For more information refer to the *Extreme Network OS IP Fabrics Configuration Guide*.

Static and automatic neighbor peer transition

- When a static neighbor is configured and the same peer is then automatically discovered through LLDP, the static neighbor configuration takes precedence over the dynamic peer configuration for as long as the static configuration exists. Once the static configuration is removed, the dynamic peer is automatically created.
- When a static neighbor is configured and the same automatically-discovered peer already exists, then the dynamic peer is reset and a new TCP session is initiated with the static configuration. When the static neighbor configuration is subsequently removed, then the dynamic neighbor is automatically created; that is, once the static configuration is removed, the dynamic peer is created.

Enabling BGP automatic neighbor discovery

BGP automatic neighbor discovery can be configured on a routing device so that only the local AS and automatic neighbor discovery need to be configured on one of the BGP peer groups to bring BGP up in an IP fabric.

BGP auto neighbor discovery is enabled only when a local AS is configured and the **neighbor accept-lldp-neighbors** command is set for one peer group (on which an MD5 password is configured).

NOTE
For security purposes, the peer group must have a Message Digest 5 (MD5) password. The MD5 password should be the same across all nodes.

NOTE
Currently, **neighbor accept-lldp-neighbors** configuration is supported for only one peer group.

Perform the following task to enable BGP automatic neighbor discovery.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   device(config)# router bgp
   ```

4. Specify the number of the autonomous system (AS) in which your device resides.

```
device(config-bgp-router)# local-as 100
```

5. Configure a neighbor peer group.

```
device(config-bgp-router)# neighbor lldp-grp peer-group
```

   This example configures a neighbor peer group named lldp-grp.

6. Enable BGP automatic neighbor discovery.

```
device(config-bgp-router)# neighbor lldp-grp accept-lldp-neighbors
```

7. Specify an MD5 password for the peer group.

```
device(config-bgp-router)# neighbor lldp-grp password mysecurepassword
```

The following example enables BGP automatic neighbor discovery for a peer group and specifies that the peer group number is 1. It also configures an MD5 password for the peer group.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 100
device(config-bgp-router)# neighbor lldp-grp peer-group
device(config-bgp-router)# neighbor lldp-grp accept-lldp-neighbors 1
device(config-bgp-router)# neighbor lldp-grp password mysecurepassword

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 100
device(config-bgp-router)# neighbor lldp-grp peer-group
device(config-bgp-router)# neighbor lldp-grp accept-lldp-neighbors
device(config-bgp-router)# neighbor lldp-grp password mysecurepassword
```

Disabling the advertisement of TLV values for automatically discovered BGP neighbors

Link Layer Discovery Protocol (LLDP) is can be configured to disable the advertisement of the Type, Length, and Values (TLV) values for automatically discovered BGP neighbors. The advertisement of the TLV values for automatically discovered BGP neighbors is enabled by default as long the feature is configured through BGP. The following task disables the advertising of BGP neighbor TLV in LLDP.

1. Enter the **configure terminal** command to access global configuration mode.

   LLDP can be configured to disable advertisements of bgp-auto-nbr-tlvs either in global LLDP mode or per interface.

```
device# configure terminal
```

2. Enter the **protocol lldp** command to enter LLDP configuration mode.

```
device(config)# protocol lldp
```

3. Enter the **no advertise bgp-auto-nbr-tlv** command to enable the advertising of BGP neighbor TLV in LLDP.

```
device(conf-lldp)# advertise bgp-auto-nbr-tlv
```

The following example disables the advertising of BGP neighbor TLV in LLDP.

```
device# configure terminal
device(config)# protocol lldp
device(conf-lldp)# no advertise bgp-auto-nbr-tlv
```

Disabling BGP auto neighbor discovery in LLDP at global level

By default, BGP auto neighbor discovery is enabled on all Link Layer Discovery Protocol (LLDP) supported interfaces. BGP auto neighbor discovery can be disabled in LLDP at either global or interface level.

Perform the following steps to disable BGP auto neighbor discovery in LLDP at the global level.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter LLDP configuration mode.

   ```
   device(config)# protocol lldp
   ```

3. Disable BGP auto neighbor discovery in LLDP.

   ```
   device(conf-lldp)# no advertise bgp-auto-nbr-tlv
   ```

   This example stops LLDP advertising the BGP auto neighbor discovery Type Length Value (TLV).

The following example disables BGP auto neighbor discovery by stopping the advertisement of BGP neighbor TLV in LLDP at a global level.

```
device# configure terminal
device(config)# protocol lldp
device(conf-lldp)# no advertise bgp-auto-nbr-tlv
```

Enabling the advertising of BGP neighbor TLV at interface level

Link Layer Discovery Protocol (LLDP) can be configured to advertise the Type, Length, and Values (TLV) values for automatically discovered BGP neighbors. The following task enables the advertising of BGP neighbor TLV for a specific LLDP profile for a specified 40-gigabit Ethernet interface.

> NOTE
> IF LLDP is enabled at the global level, it is automatically enabled on all interfaces. Therefore, this configuration is not mandatory and is only applicable where LLDP is not enabled at the global level.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **protocol lldp** command to enter LLDP configuration mode.

   ```
   device(config)# protocol lldp
   ```

3. Enter the **profile** command and specify a name to create a LLDP profile.

   ```
   device(conf-lldp)# profile test1
   ```

4. Enter the **advertise bgp-auto-nbr-tlv** command to enable the advertising of BGP neighbor TLV for the configured LLDP profile.

   ```
   device(conf-profile-test1)# advertise bgp-auto-nbr-tlv
   ```

5. Enter the **exit** command until you reach global configuration mode.

   ```
   device(conf-profile-test1)# exit
   device(conf-lldp)# exit
   ```

6. Enter the **interface** command, specifying an interface, to access interface configuration mode.

```
device(config)# interface fortygigabitethernet 1/0/49
```

7. Enter the **lldp profile** command to apply the configured LLDP profile to the interface.

```
device (conf-if-fo-1/0/49)# lldp profile 1
```

The following example enables the advertising of BGP neighbor TLV for a specific LLDP profile for a specified 40-gigabit Ethernet interface.

```
device# configure terminal
device(config)# protocol lldp
device(conf-lldp)# profile test1
device(conf-profile-test1)# advertise bgp-auto-nbr-tlv
device(conf-profile-test1)# exit
device(conf-lldp)# exit
device(config)# interface fortygigabitethernet 1/0/49
device (conf-if-fo-1/0/49)# lldp profile 1
```

Disabling BGP auto neighbor discovery in LLDP at interface level

By default, BGP auto neighbor discovery is enabled on all Link Layer Discovery Protocol (LLDP) supported interfaces. BGP auto neighbor discovery can be disabled in LLDP at either global or interface level.

BGP auto neighbor discovery is disabled at interface level by attaching an LLDP profile (on which the advertisement of BGP auto neighbor discovery Type Length Value (TLV) has been disabled) to the specific interface.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enter LLDP configuration mode.

```
device(config)# protocol lldp
```

3. Enter LLDP profile configuration mode.

```
device(config-lldp)# profile 1
```

   This example enters configuration mode for LLDP profile numbered 1.

4. Disable BGP auto neighbor discovery on the profile.

```
device(config-profile-1)# no advertise bgp-auto-nbr-tlv
```

   This example stops the advertisement the BGP auto neighbor discovery Type Length Value (TLV) for LLDP profile 1.

5. Return to global configuration mode.

```
device(config-profile-1)# end
```

6. Enter configuration mode for the interface on which BGP auto neighbor discovery is to be disabled.

```
device(config)# interface ethernet 0/1
```

7. To disable BGP auto neighbor discovery on this interface, attach the profile on which BGP auto neighbor discovery TLV is disabled, to the interface.

```
device(conf-if-eth-0/1)# profile 1
```

The following example shows how to disable the advertisement of BGP neighbor TLV in LLDP on Ethernet interface 0/1.

```
device# configure terminal
device(config)# protocol lldp
device(conf-lldp)# profile 1
device(config-profile-1)# no advertise bgp-auto-nbr-tlv
device(config-profile-1)# end
!
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# profile 1
```

Generalized TTL Security Mechanism support

Generalized TTL Security Mechanism (GTSM) is a lightweight security mechanism that protects external Border Gateway Protocol (eBGP) peering sessions from CPU utilization-based attacks using forged IP packets. GTSM prevents attempts to hijack the eBGP peering session by a host on a network segment that is not part of either BGP network, or by a host on a network segment that is not between the eBGP peers.

GTSM is enabled by configuring a minimum Time To Live (TTL) value for incoming IP packets received from a specific eBGP peer. BGP establishes and maintains the session only if the TTL value in the IP packet header is equal to or greater than the TTL value configured for the peering session. If the value is less than the configured value, the packet is silently discarded and no Internet Control Message Protocol (ICMP) message is generated.

In the case of directly connected neighbors, the device expects the BGP control packets received from the neighbor to have a TTL value of either 254 or 255. For multihop peers, the device expects the TTL for BGP control packets received from the neighbor to be greater than or equal to 255, minus the configured number of hops to the neighbor. If the BGP control packets received from the neighbor do not have the expected value, the device drops them.

When GTSM protection is enabled, BGP control packets sent by the device to a neighbor have a Time To Live (TTL) value of 255. In addition, the device expects the BGP control packets received from the neighbor to have a TTL value of either 254 or 255. For multihop peers, the device expects the TTL for BGP control packets received from the neighbor to be greater than or equal to 255, minus the configured number of hops to the neighbor. If the BGP control packets received from the neighbor do not have the anticipated value, the device drops them.

For more information on GTSM protection, refer to RFC 3682.

Assumptions and limitations

- GTSM is supported for both directly connected peering sessions and multihop eBGP peering sessions.
- GTSM is supported for eBGP only.
- GTSM does not protect the integrity of data sent between eBGP peers and does not validate eBGP peers through any authentication method.
- GTSM validates only the locally configured TTL count against the TTL field in the IP packet header.
- GTSM should be configured on each participating device to maximize the effectiveness of this feature.
- When GTSM is enabled, the eBGP session is secured in the incoming direction only and has no effect on outgoing IP packets or the remote device.

Configuring GTSM for BGP4

Generalized TTL Security Mechanism (GTSM) can be configured to protect external Border Gateway Protocol (eBGP) peering sessions.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   device(config-bgp)# local-as 65520
   ```

5. Enter the **neighbor remote-as** command to add a neighbor.

   ```
   device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
   device(config-bgp)# neighbor 10.10.10.1 remote-as 2
   ```

6. Enter the **neighbor ebgp-btsh** command, specifying an IP address, to enable GTSM.

   ```
   device(config-bgp-router)# neighbor 10.10.10.1 ebgp-btsh
   device(config-bgp)# neighbor 10.10.10.1 ebgp-btsh
   ```

The following example enables GTSM between a device and a neighbor with the IP address 10.10.10.1.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
device(config-bgp-router)# neighbor 10.10.10.1 ebgp-btsh

device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# neighbor 10.10.10.1 remote-as 2
device(config-bgp)# neighbor 10.10.10.1 ebgp-btsh

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
device(config-bgp-router)# neighbor 10.10.10.1 ebgp-btsh
```

Disabling the BGP AS_PATH check function

A device can be configured so that the AS_PATH check function for routes learned from a specific location is disabled, and routes that contain the recipient BGP speaker's AS number are not rejected.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

   ```
   device(config)# router bgp
   ```

4. Enter the **address-family ipv4 unicast** command to enter BGP IPv4 address family unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

5. Enter the **neighbor allowas-in** command and specify a **number** to disable the BGP AS_PATH check function, and specify the number of times that the AS path of a received route may contain the recipient BGP speaker's AS number and still be accepted.

   ```
   device(config-bgp)# neighbor 10.1.1.1 allowas-in 3
   ```

   ```
   device(config-bgp-ipv6u)# neighbor 10.1.1.1 allowas-in 3
   ```

The following example specifies that the AS path of a received route may contain the recipient BGP speaker's AS number three times and still be accepted.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.1.1 allowas-in 3

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.1.1 allowas-in 3

device# configure terminal
device(config)# router bgp
device(config-bgp)# neighbor 10.1.1.1 allowas-in 3
```

Using route maps

A route map is a named set of match conditions and parameter settings that the device can use to modify route attributes and to control redistribution of the routes into other protocols. A route map consists of a sequence of instances, the equivalent of rows in a table. The device evaluates a route according to route map instances in ascending numerical order. The route is first compared against instance 1, then against instance 2, and so on. When a match is found, the device stops evaluating the route.

Route maps can contain **match** clauses and **set** statements. Each route map contains a **permit** or **deny** statement for routes that match the **match** clauses:

Route maps can contain **match** clauses and **set** statements. Each route map contains a **permit** or **deny** statement for routes that match the **match** clauses (except when a **continue** statement is used):

- If the route map contains a **permit** statement, a route that matches a match statement is permitted; otherwise, the route is denied.
- If the route map contains a **deny** statement, a route that matches a match statement is denied.
- If a route does not match any **match** statements in the route map, then the route is denied. This is the default action. To change the default action, configure the last **match** statement in the last instance of the route map to **permit any any** .
- If there is no **match** statement, the software considers the route to be a match.
- For route maps that contain address filters, AS-path filters, or community filters, if the action specified by a filter conflicts with the action specified by the route map, the route map action takes precedence over the filter action.

If the route map contains **set** statements, routes that are permitted by the route map **match** statements are modified according to the **set** statements.

Match statements

Match statements compare the route against one or more of the following:

- The route BGP4 MED (metric)
- The IP address of the next hop device
- The route tag
- For OSPF routes only, the route type (internal, external type 1, or external type 2)
- An AS-path access control list (ACL)
- A community ACL
- An IP prefix list
- An extended-community-based ACL

Set statements

For routes that match all of the **match** statements, for all the routes that are permitted by the route-map match statements, the route map **set** statements can perform one or more of the following modifications to the route attributes:

- Prepend AS numbers to the front of the route AS-path. By adding AS numbers to the AS-path, you can cause the route to be less preferred when compared to other routes based on the length of the AS-path.
- Add a user-defined tag or an automatically calculated tag to the route.
- Set the community attributes.
- Set the extended community attributes.
- Set the local preference.
- Set the MED (metric).
- Set the IP address of the next-hop device.
- Set the origin to IGP or INCOMPLETE.
- Set the weight.

When you configure parameters for redistributing routes into BGP4, one of the optional parameters is a route map. If you specify a route map as one of the redistribution parameters, the device matches the route against the match statements in the route map. If a match is found and if the route map contains **set** statements, the device sets the attributes in the route according to the set statements.

Refer to the following commands in the *Extreme Network OS Command Reference* for more information on creating and using route maps:

Refer to the following commands in the *Extreme SLX-OS Command Reference* for more information on creating and using route maps:

- continue
- match (route-map)
- route-map
- set as-path prepend

Matching on an AS-path

A route map that matches on a specified AS-path access control list (ACL) can be configured.

An AS-path ACL must be configured using the **ip as-path access-list** command.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 5
   ```

3. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

   ```
   device(config)# route-map myaclroutemap1 permit 10
   ```

   ```
   device(config-rbridge-id-5)# route-map myaclroutemap1 permit 10
   ```

4. Enter the **match** command and specify an ACL name to configure the route map to match on the specified AS-path ACL.

   ```
   device(config-routemap myaclroutemap1)# match as-path myaspath
   ```

   ```
   device(config-route-map-myaclroutemap1/permit/10)# match as-path myaspath
   ```

The following example configures a route map instance that matches on AS-path ACL “myaspath”.

```
device# configure terminal
device(config)# route-map myaclroutemap1 permit 10
device(config-routemap myaclroutemap1)# match as-path myaspath

device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# route-map myaclroutemap1 permit 10
device(config-route-map-myaclroutemap1/permit/10)# match as-path myaspath

device# configure termnial
device(config)# route-map myaclroutemap1 permit 10
device(config-route-map-myaclroutemap1/permit/10)# match as-path myaspath
```

Matching on a community ACL

A route map instance that matches on a specified community access control list (ACL) can be configured.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 5
```

3. Enter the **ip community-list standard** command, specifying a community number and using the **permit** parameter to create a community ACL that permits traffic.

```
device(config)# ip community-list standard 1 permit 123:2
```

```
device(config-rbridge-id-5)# ip community-list standard 1 permit 123:2
```

4. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

```
device(config)# route-map mycommroutemap1 permit 10
```

```
device(config-rbridge-id-5)# route-map mycommroutemap1 permit 10
```

5. Enter the **match community** command, and specify a community number to match the BGP community access list name in the configured route-map instance.

```
device(config-route-map-mycommroutemap1)# match community 1
```

```
device(config-route-map-mycommroutemap1/permit/10)# match community 1
```

The following example matches community ACL "1" for route map instance "mycommroutemap1".

```
device# configure terminal
device(config)# ip community-list standard 1 permit 123:2
device(config)# route-map mycommroutemap1 permit 10
device(config-route-map-mycommroutemap1)# match community 1

device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# ip community-list standard 1 permit 123:2
device(config-rbridge-id-5)# route-map mycommroutemap1 permit 10
device(config-route-map-mycommroutemap1/permit/10)# match community 1

device# configure terminal
device(config)# ip community-list standard 1 permit 123:2
device(config)# route-map mycommroutemap1 permit 10
device(config-route-map-mycommroutemap1/permit/10)# match community 1
```

Matching on a destination network

A route map that matches on a destination network can be configured.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 5
```

3. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

```
device(config)# route-map mynetroutemap1 permit 10
```

```
device(config-rbridge-id-5)# route-map mynetroutemap1 permit 10
```

4. Enter the **match** command with the **ip address** parameter. Specify a prefix list using the **ip prefix-list** *string* parameter to configure the route map to match on the specified prefix.

```
device(config-route-map-mynetroutemap1)# match ip address prefix-list mylist
```

```
device(config-route-map-mynetroutemap1/permit/10)# match ip address prefix-list mylist
```

The following example configures a route map instance that matches on a specified destination network.

```
device# configure terminal
device(config)# route-map mynetroutemap1 permit 10
device(config-route-map-mynetroutemap1)# match ip address prefix-list mylist

device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# route-map mynetroutemap1 permit 10
device(config-route-map-mynetroutemap1/permit/10)# match ip address prefix-list mylist

device# configure terminal
device(config)# route-map mynetroutemap1 permit 10
device(config-route-map-mynetroutemap1/permit/10)# match ip address prefix-list mylist
```

Matching on a BGP4 static network

A route map that matches on a static network can be configured. In this task, a route-map is configured to match on the BGP4 static network. The device is then configured to filter the outgoing route updates to a specified BGP neighbor according to the set of attributes defined in the configured route map.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 5
```

3. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

```
device(config)# route-map mystaticroutemap3 permit 1
```

```
device(config-rbridge-id-5)# route-map mystaticroutemap3 permit 1
```

4. Enter the **match** command with the **protocol bgp static-network** parameter to match on BGP4 static network routes.

```
device(config-routemap-mystaticroutemap3)# match protocol bgp static-network
```

```
device(config-routemap-mystaticroutemap3/permit/1)# match protocol bgp static-network
```

5. Enter the **set local-preference** command and enter a value to set a BGP local-preference path attribute in the route-map instance.

```
device(config-routemap-mystaticroutemap3/)# set local-preference 150
```

```
device(config-routemap-mystaticroutemap3/permit/1)# set local-preference 150
```

6. Enter the **set community** command with the **no export** parameter to set the BGP community attribute for the route-map instance not to export to the next AS.

```
device(config-routemap-mystaticroutemap3)# set community no-export
device(config-routemap-mystaticroutemap3/permit/1)# set community no-export
```

7. Enter the **exit** command to exit route map configuration mode.

```
device(config-routemap-mystaticroutemap3)# exit
device(config-routemap-mystaticroutemap3/permit/1)# exit
```

8. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
device(config)# router bgp
```

9. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

10. Enter the **neighbor** *ip-address* **route-map** command and specify the **out** keyword, specifying the route map name, to filter the outgoing route updates to a specified BGP neighbor according to the set of attributes defined in the configured route map.

```
device (config-bgp)# neighbor 10.1.2.3 route-map out mystaticroutemap3
device (config-bgp-ipv4u)# neighbor 10.1.2.3 route-map out mystaticroutemap3
```

The following example configures a route map instance that matches on a BGP4 static network and configures the device to filter the outgoing route updates to a specified BGP neighbor according to the set of attributes defined in the configured route map.

```
device# configure terminal
device(config)# route-map mystaticroutemap3 permit 1
device(config-routemap-mystaticroutemap3)# match protocol bgp static-network
device(config-routemap-mystaticroutemap3)# set local-preference 150
device(config-routemap-mystaticroutemap3)# set community no-export
device(config-routemap-mystaticroutemap3)# exit
device(config)# router bgp
device(config-bgp)# neighbor 10.1.2.3 route-map out mystaticroutemap3

device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# route-map mystaticroutemap3 permit 1
device(config-routemap-mystaticroutemap3/permit/1)# match protocol bgp static-network
device(config-routemap-mystaticroutemap3/permit/1)# set local-preference 150
device(config-routemap-mystaticroutemap3/permit/1)# set community no-export
device(config-routemap-mystaticroutemap3/permit/1)# exit
device(config-rbridge-id-5)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map out mystaticroutemap3

device# configure terminal
device(config)# route-map mystaticroutemap3 permit 1
device(config-routemap-mystaticroutemap3/permit/1)# match protocol bgp static-network
device(config-routemap-mystaticroutemap3/permit/1)# set local-preference 150
device(config-routemap-mystaticroutemap3/permit/1)# set community no-export
device(config-routemap-mystaticroutemap3/permit/1)# exit
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map out mystaticroutemap3
```

Matching on a next-hop device

A route map that matches on a next-hop device can be configured.

A prefix list must be configured using the **ip prefix-list** command.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 5
   ```

3. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

   ```
   device(config)# route-map myhoproutemap1 permit 10
   device(config-rbridge-id-5)# route-map myhoproutemap1 permit 10
   ```

4. Enter the **match** command, using the **next-hop** parameter and specify a prefix-list, to match IP next-hop match conditions for a specified prefix list in a route-map instance.

   ```
   device(config-routemap myaclroutemap1)# match ip next-hop    prefix-list mylist
   device(config-route-map-myhoproutemap1/permit/10)# match ip next-hop    prefix-list mylist
   ```

The following example configures a route map and specifies a prefix list to match on a next-hop device.

```
device# configure terminal
device(config)# route-map myhoproutemap1 permit 10
device(config-route-map-myhoproutemap1)# match ip next-hop    prefix-list mylist

device# configure terminal
device(config)# rbridge-id-5
device(config-rbridge-id-5)# route-map myhoproutemap1 permit 10
device(config-route-map-myhoproutemap1/permit/10)# match ip next-hop    prefix-list mylist

device# configure terminal
device(config)# route-map myhoproutemap1 permit 10
device(config-route-map-myhoproutemap1/permit/10)# match ip next-hop    prefix-list mylist
```

Matching on an interface

To configure a route map that matches on an interface:

```
device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# route-map myintroutemap1 permit 99
device(config-route-map-myintroutemap1/permit/99)# match interface tengigabitethernet 5/1/1 loopback 1

device# configure terminal
device(config)# route-map myintroutemap1 permit 99
device(config-route-map-myintroutemap1/permit/99)# match interface ethernet 1/1 loopback 1

device# configure terminal
device(config)# route-map myintroutemap1 permit 99
device(config-route-map-myintroutemap1/permit/99)# match interface ethernet 0/1 loopback 1
```

Using route-map continue statements

Continuation statements can be configured in a route map. This task configures a route map instance and adds two route-map continue statements to the route map instance.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 5
```

3. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

```
device(config)# route-map mcontroutemap1 permit 1
device(config-rbridge-id-5)# route-map mcontroutemap1 permit 1
```

4. Enter the **match** command with the metric parameter and specify a value to match a route metric in the route-map instance.

```
device(config-routemap mcontroutemap1)# match metric 10
device(config-route-map-mcontroutemap1/permit/1)# match metric 10
```

5. Enter the **set weight** command and specify a value to set a BGP weight for the routing table in the route-map instance.

```
device(config-routemap mcontroutemap1)# set weight 10
device(config-route-map-mcontroutemap1/permit/1)# set weight 10
```

6. Enter the **continue** command and specify a value to configure a route-map instance number that goes in a continue statement in the route-map instance.

```
device(config-routemap mcontroutemap1)# continue 2
device(config-routemap-mycontroutemap/permit/1)# continue 2
```

7. Enter the **exit** command to exit route map configuration mode.

```
device(config-routemap mcontroutemap1)# exit
device(config-routemap-mycontroutemap/permit/1)# exit
```

8. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

```
device(config)# route-map mcontroutemap1 permit 2
device(config-rbridge-id-5)# route-map mcontroutemap1 permit 2
```

9. Enter the **match** command with the metric parameter and specify a value to match a route metric in the route-map instance.

```
device(config-routemap mcontroutemap1)# match metric 10
device(config-route-map-mcontroutemap1/permit/2)# match metric 10
```

10. Enter the **set weight** command and specify a value to set a BGP weight for the routing table in the route-map instance.

```
device(config-routemap mcontroutemap1)# set weight 20
device(config-route-map-mcontroutemap1/permit/2)# set weight 20
```

11. Enter the **continue** command and specify a value to configure a route-map instance number that goes in a continue statement in the route-map instance.

```
device(config-routemap mcontroutemap2)# continue 3
device(config-routemap-mycontroutemap/permit/2)# continue 3
```

The following example configures a route map instance and adds two route-map continue statements to the route map instance.

```
device# configure terminal
device(config)# route-map mcontroutemap1 permit 1
device(config-routemap-mcontroutemap1)# match metric 10
device(config-routemap-mcontroutemap1)# set weight 10
device(config-routemap-mcontroutemap1)# match metric 10
device(config-routemap-mcontroutemap1)# continue 2
device(config-routemap-mcontroutemap1)# exit
device(config)# route-map mcontroutemap1 permit 2
device(config-routemap-mycontroutemap1)# match tag 10
device(config-routemap-mycontroutemap1)# set weight 20

device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# route-map mcontroutemap1 permit 1
device(config-routemap-mcontroutemap1/permit/1)# match metric 10
device(config-routemap-mcontroutemap1/permit/1)# set weight 10
device(config-routemap-mycontroutemap1/permit/1)# match metric 10
device(config-routemap-mycontroutemap1/permit/1)# continue 2
device(config-routemap-mycontroutemap1/permit/1)# exit
device(config-rbridge-id-5)# route-map mcontroutemap1 permit 2
device(config-routemap-mycontroutemap1/permit/2)# match tag 10
device(config-routemap-mycontroutemap1/permit/2)# set weight 20

device# configure terminal
device(config)# route-map mcontroutemap1 permit 1
device(config-routemap-mycontroutemap1/permit/1)# match metric 10
device(config-routemap-mycontroutemap1/permit/1)# set weight 10
device(config-routemap-mycontroutemap1/permit/1)# match metric 10
device(config-routemap-mycontroutemap1/permit/1)# continue 2
device(config-routemap-mycontroutemap1/permit/1)# exit
device(config)# route-map mcontroutemap1 permit 2
device(config-routemap-mycontroutemap1/permit/2)# match tag 10
device(config-routemap-mycontroutemap1/permit/2)# set weight 20
```

Route-map continue statement for BGP4 routes

A continue statement in a route-map directs program flow to skip over route-map instances to another, user-specified instance. If a matched instance contains a continue statement, the system looks for the instance that is identified in the statement.

The continue statement in a matching instance initiates another traversal at the instance specified. The system records all of the matched instances and, if no deny statements are encountered, proceeds to execute the set clauses of the matched instances.

If the system scans all route-map instances but finds no matches, or if a deny condition is encountered, then it does not update the routes. Whenever a matched instance contains a deny statement, the current traversal terminates, and none of the updates specified in the set statements of the matched instances in both current and previous traversals are applied to the routes.

This supports a more programmable route-map configuration and route filtering scheme for BGP4 peering. It can also execute additional instances in a route map after an instance is executed by means of successful match statements. You can configure and organize more-modular policy definitions to reduce the number of instances that are repeated within the same route map.

This feature currently applies to BGP4 routes only. For protocols other than BGP4, continue statements are ignored.

Using a route map to configure dampening

You can set a BGP route-flap dampening penalty in a route-map instance..

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 5
   ```

3. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

   ```
   device(config)# route-map myroutemap permit 1
   ```

   ```
   device(config-rbridge-id-5)# route-map myroutemap permit 1
   ```

4. Enter the **set dampening** command and specify a value name to set the BGP route-flap dampening penalty for the route-map instance.

   ```
   device(config-route-map-myroutemap)# set dampening 20
   ```

   ```
   device(config-route-map-myroutemap/permit/1)# set dampening 20
   ```

The following example configures a route map instance and sets a BGP route-flap dampening penalty of 20.

```
device# configure terminal
device(config)# route-map myroutemap permit 1
device(config-route-map-myroutemap)# set dampening 20

device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# route-map myroutemap permit 1
device(config-route-map-myroutemap/permit/1)# set dampening 20

device# configure termnial
device(config)# route-map myroutemap permit 1
device(config-route-map-myroutemap/permit/1)# set dampening 20
```

Clearing diagnostic buffers

The device stores the following BGP4 diagnostic information in buffers:

- The first 400 bytes of the last packet received that contained an error
- The last NOTIFICATION message either sent or received by the device

This information can be useful if you are working with Extreme Technical Support to resolve a problem. The buffers do not identify the system time when the data was written to the buffer. If you want to ensure that diagnostic data in a buffer is recent, you can clear the buffers. You can clear the buffers for a specific neighbor or for all neighbors.

If you clear the buffer containing the first 400 bytes of the last packet that contained errors, all the bytes are changed to zeros. The Last Connection Reset Reason field of the BGP4 neighbor table also is cleared.

If you clear the buffer containing the last NOTIFICATION message sent or received, the buffer contains no data.

You can clear the buffers for all neighbors, for an individual neighbor, or for all the neighbors within a specific peer group.

Refer to the NetIron Command Reference for more information.

Refer to the *Extreme Network OS Command Reference* for more information.

Refer to the *Extreme SLX-OS Command Reference* for more information.

Displaying BGP4 statistics

Various **show ip bgp** commands verify information about BGP4 configurations.

Use one or more of the following commands to verify BGP4 information. The commands do not have to be entered in this order.

1. Enter the **show ip bgp summary** command.

```
device> show ip bgp summary

  BGP4 Summary
  Router ID: 7.7.7.7   Local AS Number: 100
  Confederation Identifier: not configured
  Confederation Peers:
  Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
  Number of Neighbors Configured: 1, UP: 1
  Number of Routes Installed: 0
  Number of Routes Advertising to All Neighbors: 0 (0 entries)
  Number of Attribute Entries Installed: 0
  '+': Data in InQueue '>': Data in OutQueue '-': Clearing
  '*': Update Policy 'c': Group change 'p': Group change Pending
  'r': Restarting 's': Stale '^': Up before Restart '<': EOR waiting
  Neighbor Address  AS#         State   Time          Rt:Accepted Filtered Sent     ToSend
  10.1.1.8          100         ESTAB   0h 9m16s      0           0        0        0
```

   This example output gives summarized BGP4 information.

2. Enter the **show ip bgp routes** command.

```
device> show ip bgp routes

Total number of BGP Routes: 2000
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE x:BEST-EXTERNAL
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      150.150.150.0/24   103.103.1.1     2          100        0      BEx
         AS_PATH: 201
2      150.150.150.0/24   103.103.2.1     3          100        0      E
         AS_PATH: 202
3      150.150.150.0/24   103.103.3.1     4          100        0      E
         AS_PATH: 203
4      150.150.150.0/24   103.103.4.1     5          100        0      E
         AS_PATH: 204
5      150.150.150.0/24   103.103.5.1     6          100        0      E
…
```

   This example shows general BGP4 route information.

3. Enter the **show ip bgp** command.

```
device> show ip bgp

Total number of BGP Routes: 4
Status codes: s suppressed, d damped, h history, * valid, > best, i internal, S stale, x best-
external
Origin codes: i - IGP, e - EGP, ? - incomplete
    Network            Next Hop        MED        LocPrf     Weight Path
*>i 110.110.110.0/24   50.50.50.10                150        0      i
*x  110.110.110.0/24   20.20.20.10                100        0      200 i
*   110.110.110.0/24   30.30.30.10                100        0      300 i
*   110.110.110.0/24   40.40.40.10                100        0      400 i
```

This example shows general BGP4 information.

4. Enter the **show ip bgp attribute-entries** command.

```
device> show ip bgp attribute-entries

        Total number of BGP Attribute Entries: 18 (0)
1       Next Hop  :192.168.1.6        MED :1             Origin:INCOMP
        Originator:0.0.0.0            Cluster List:None
        Aggregator:AS Number :0      Router-ID:0.0.0.0     Atomic:None
        Local Pref:100               Communities:Internet
        Extended Community: SOO 300000:3
        AS Path   :90000 80000 (length 11)
        Address: 0x10e4e0c4  Hash:489 (0x03028536), PeerIdx 0
        Links: 0x00000000, 0x00000000, nlri: 0x10f4804a
        Reference Counts: 1:0:1, Magic: 51
2       Next Hop  :192.168.1.5        Metric   :1         Origin:INCOMP
        Originator:0.0.0.0            Cluster List:None
        Aggregator:AS Number :0      Router-ID:0.0.0.0     Atomic:None
        Local Pref:100               Communities:Internet
        Extended Community: RT 200000:2
        AS Path   :90000 75000 (length 11)
        Address: 0x10e4e062  Hash:545 (0x0301e8f6), PeerIdx 0
        Links: 0x00000000, 0x00000000, nlri: 0x10f47ff0
        Reference Counts: 1:0:1, Magic: 49
```

This example shows information about one route-attribute entry that is stored in device memory.

5. Enter the **show ip bgp peer-group** command.

```
device# show ip bgp peer-group STR

1   BGP peer-group is STR
      Address family : IPV4 Unicast
        activate
      Address family : IPV4 Multicast
        no activate
      Address family : IPV6 Unicast
        no activate
      Address family : IPV6 Multicast
        no activate
      Address family : VPNV4 Unicast
        no activate
      Address family : L2VPN VPLS
        no activate
    Members:
       IP Address: 10.1.1.1, AS: 5
```

This example shows output for one BGP peer group, called “STR”.

6. Enter the **show ip bgp routes** command using the **summary** keyword.

```
device> show ip bgp routes summary

  Total number of BGP routes (NLRIs) Installed     : 20
  Distinct BGP destination networks                : 20
  Filtered BGP routes for soft reconfig            : 100178
  Routes originated by this router                 : 2
  Routes selected as BEST routes                   : 19
  BEST routes not installed in IP forwarding table : 1
  Unreachable routes (no IGP route for NEXTHOP)    : 1
  IBGP routes selected as best routes              : 0
  EBGP routes selected as best routes              : 17
```

This example shows summarized BGP4 route information.

Displaying BGP4 statistics

Various **show ip bgp** commands verify information about BGP4 configurations.

Use one or more of the following commands to verify BGP4 information. The commands do not have to be entered in this order.

1. Enter the **show ip bgp summary** command.

```
device# show ip bgp summary

BGP4 Summary
Router ID: 10.10.1.4   Local AS Number: 200
Confederation Identifier: not configured
Confederation Peers:
Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
Number of Neighbors Configured: 67, UP: 67
Number of Routes Installed: 258088, Uses 22195568 bytes
Number of Routes Advertising to All Neighbors: 17,035844(3,099146 entries), Uses 192,147052 bytes
Number of Attribute Entries Installed: 612223, Uses 55100070 bytes
Neighbor Address  AS#         State     Time     Rt:Accepted Filtered Sent ToSend
10.1.1.1         100         CONN      1h 3m16s    0         0        0    0
10.1.1.2         100         CONN      1h 3m16s    0         0        0    0
10.1.1.3         100         CONN      1h 3m16s    0         0        0    0
10.1.1.4         100         CONN      1h 3m16s    0         0        0    0
10.1.1.5         100         CONN      1h 3m16s    0         0        122  0
10.1.1.6         100         CONN      1h 3m16s    0         0        122  0
10.1.1.7         100         CONN      1h 3m16s    0         0        0    2
```

This example output gives summarized BGP4 information.

2. Enter the **show ip bgp routes** command, using the **summary** keyword.

```
device# show ip bgp routes summary
  Total number of BGP routes (NLRIs) Installed     : 20
  Distinct BGP destination networks                : 20
  Filtered BGP routes for soft reconfig            : 100178
  Routes originated by this router                 : 2
  Routes selected as BEST routes                   : 19
  BEST routes not installed in IP forwarding table : 1
  Unreachable routes (no IGP route for NEXTHOP)    : 1
  IBGP routes selected as best routes              : 0
  EBGP routes selected as best routes              : 17
```

This example shows summarized BGP4 route information.

3. Enter the **show ip bgp routes** command.

```
device# show ip bgp routes
Total number of BGP Routes: 26
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
      Prefix             Next Hop        MED       LocPrf     Weight Status
1     10.3.0.0/8         192.168.4.106    1         100        0      BE
         AS_PATH: 65001 4355 701 80
2     10.4.0.0/8         192.168.4.106    107       100        0      BE
         AS_PATH: 65001 4355 1
3     10.60.212.0/22     192.168.4.106    107       100        0      BE
         AS_PATH: 65001 4355 701 1 189
4     10.6.0.0/8         192.168.4.106    107       100        0      BE
         AS_PATH: 65001 4355 3356 7170 1455
5     10.8.1.0/24        192.168.4.106    0         100        0      BE
         AS_PATH: 65001
```

This example shows general BGP4 route information.

4. Enter the **show ip bgp routes** command, using the **unreachable** keyword.

```
device# show ip bgp routes unreachable
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED     LocPrf     Weight Status
1      10.8.8.0/24        192.168.5.1     0          101        0
         AS_PATH: 65001 4355 1
```

This example shows BGP4 routes whose destinations are unreachable using any of the BGP4 paths in the BGP4 route table.

5. Enter the **show ip bgp flap-statistics** command.

```
device# show ip bgp flap-statistics
Total number of flapping routes: 414
    Status Code  >:best d:damped h:history *:valid
    Network            From         Flaps Since    Reuse    Path
h>  10.50.206.0/23   10.90.213.77   1     0 :0 :13 0 :0 :0  65001 4355 1 701
h>  10.255.192.0/20  10.90.213.77   1     0 :0 :13 0 :0 :0  65001 4355 1 7018
h>  10.252.165.0/24  10.90.213.77   1     0 :0 :13 0 :0 :0  65001 4355 1 7018
h>  10.50.208.0/23   10.90.213.77   1     0 :0 :13 0 :0 :0  65001 4355 1 701
h>  10.33.0.0/16     10.90.213.77   1     0 :0 :13 0 :0 :0  65001 4355 1 701
*>  10.17.220.0/24   10.90.213.77   1     0 :1 :4  0 :0 :0  65001 4355 701 62
```

This example shows the routes in the BGP4 route table that have changed state and have been marked as flapping routes.

Displaying BGP4 statistics

Various **show ip bgp** commands verify information about BGP4 configurations.

Use one or more of the following commands to verify BGP4 information. The commands do not have to be entered in this order.

1. Enter the **show ip bgp summary** command.

```
device# show ip bgp summary

  BGP4 Summary
  Router ID: 192.117.117.36   Local AS Number: 1001
  Confederation Identifier: not configured
  Confederation Peers:
  Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
  Number of Neighbors Configured: 1, UP: 0
  Number of Routes Installed: 4, Uses 384 bytes
  Number of Routes Advertising to All Neighbors: 4 (4 entries), Uses 240 bytes
  Number of Attribute Entries Installed: 1, Uses 104 bytes
  Neighbor Address  AS#         State     Time     Rt:Accepted Filtered Sent     ToSend
  192.117.117.3     1001        ACTIV     4h 5m 0s    0        0        0        4
```

   This example output gives summarized BGP4 information.

2. Enter the **show ip bgp routes** command.

```
device# show ip bgp routes

Total number of BGP Routes: 4
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      42.9.1.0/24        0.0.0.0         0          100        32768  BL
         AS_PATH:
2      42.9.2.0/24        0.0.0.0         0          100        32768  BL
         AS_PATH:
3      42.9.3.0/24        0.0.0.0         0          100        32768  BL
         AS_PATH:
4      42.9.4.0/24        0.0.0.0         0          100        32768  BL
         AS_PATH:
```

   This example shows general BGP4 route information.

3. Enter the **show ip bgp** command.

```
device# show ip bgp

Total number of BGP Routes: 4
Status codes: s suppressed, d damped, h history, * valid, > best, i internal, S stale
Origin codes: i - IGP, e - EGP, ? - incomplete
    Network            Next Hop        MED        LocPrf     Weight Path
*>  42.9.1.0/24        0.0.0.0         0          100         32768  i
*>  42.9.2.0/24        0.0.0.0         0          100         32768  i
*>  42.9.3.0/24        0.0.0.0         0          100         32768  i
*>  42.9.4.0/24        0.0.0.0         0          100         32768  i
```

   This example shows general BGP4 information.

4. Enter the **show ip bgp attribute-entries** command.

```
device# show ip bgp attribute-entries

        Total number of BGP Attribute Entries: 1
1       Next Hop  : 0.0.0.0            MED      :0               Origin:IGP
        Originator:0.0.0.0            Cluster List:None
        Aggregator:AS Number :0            Router-ID:0.0.0.0          Atomic:None
        Local Pref:100               Communities:Internet
        AS Path   : (length 0)
           AsPathLen: 0  AsNum: 0, SegmentNum: 0, Neighboring As: 0, Source As 0
        Address: 0x67c1002c  Hash:364 (0x01000000)
        Links: 0x00000000, 0x00000000
        Reference Counts: 4:0:4, Magic: 1
```

This example shows information about one route-attribute entry that is stored in device memory.

5. Enter the **show ip bgp peer-group** command.

```
device# show ip bgp peer-group

1   BGP peer-group is pg
      Address family : IPV4 Unicast
        activate
      Address family : IPV6 Unicast
        no activate
    Members:
       IP Address: 1.1.1.1, AS: 100
       IP Address: 1::1, AS: 100

2   BGP peer-group is pg6
      Address family : IPV4 Unicast
        activate
      Address family : IPV6 Unicast
        no activate
    Currently there are no members
```

This example shows output for two BGP peer groups, called “pg” and “pg6”.

6. Enter the **show ip bgp routes** command using the **summary** keyword.

```
device# show ip bgp routes summary

  Total number of BGP routes (NLRIs) Installed     : 1
  Distinct BGP destination networks                : 1
  Filtered bgp routes for soft reconfig            : 0
  Routes originated by this router                 : 1
  Routes selected as BEST routes                   : 1
  Routes Installed as BEST routes                  : 1
  BEST routes not installed in IP forwarding table : 0
  Unreachable routes (no IGP route for NEXTHOP)    : 0
  IBGP routes selected as best routes              : 0
  EBGP routes selected as best routes              : 0
  BEST routes not valid for IP forwarding table    : 0
```

This example shows summarized BGP4 route information.

Displaying BGP4 statistics

Various **show ip bgp** commands verify information about BGP4 configurations.

Use one or more of the following commands to verify BGP4 information. The commands do not have to be entered in this order.

1. Enter the **show ip bgp summary** command.

```
device# show ip bgp summary

 BGP4 Summary
  Router ID: 1.1.1.1   Local AS Number: 100
  Confederation Identifier: not configured
  Confederation Peers:
  Maximum Number of IP ECMP Paths Supported for Load Sharing: 64
  Number of Neighbors Configured: 65, UP: 65
  Number of Routes Installed: 491790, Uses 47211840 bytes
  Number of Routes Advertising to All Neighbors: 964418 (45204 entries), Uses 2712240 bytes
  Number of Attribute Entries Installed: 130, Uses 13520 bytes
  Neighbor Address  AS#         State     Time     Rt:Accepted Filtered Sent     ToSend
  52.0.0.2          2           ESTAB     5h57m22s    7500     0        7568     0
  53.0.0.1          200         ESTAB     5h57m15s    7566     0        15068    0
  53.0.1.1          200         ESTAB     5h57m15s    7566     0        15068    0
  53.0.2.1          200         ESTAB     5h57m15s    7566     0        15068    0
  53.0.3.1          200         ESTAB     5h57m15s    7566     0        15068    0
  53.0.4.1          200         ESTAB     5h57m15s    7566     0        15068    0
  53.0.5.1          200         ESTAB     5h57m15s    7566     0        15068    0
  53.0.6.1          200         ESTAB     5h57m15s    7566     0        15068    0
  53.0.7.1          200         ESTAB     5h57m15s    7566     0        15068    0
  53.0.8.1          200         ESTAB     5h57m16s    7566     0        15068    0
  53.0.9.1          200         ESTAB     5h57m15s    7566     0        15068    0
  …
```

This example output gives summarized BGP4 information.

2. Enter the **show ip bgp routes** command.

```
device# show ip bgp routes

Total number of BGP Routes: 491790
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      1.1.1.1/32         0.0.0.0         0          100        32768  BL
         AS_PATH:
2      2.2.2.2/32         53.0.16.1       0          100        0      BME
         AS_PATH: 200
3      2.2.2.2/32         53.0.17.1       0          100        0      ME
         AS_PATH: 200
4      2.2.2.2/32         53.0.18.1       0          100        0      ME
         AS_PATH: 200
5      2.2.2.2/32         53.0.19.1       0          100        0      ME
         AS_PATH: 200
6      2.2.2.2/32         53.0.20.1       0          100        0      ME
         AS_PATH: 200
7      2.2.2.2/32         53.0.21.1       0          100        0      ME
         AS_PATH: 200
8      2.2.2.2/32         53.0.22.1       0          100        0      ME
         AS_PATH: 200
9      2.2.2.2/32         53.0.23.1       0          100        0      ME
         AS_PATH: 200
…
```

This example shows general BGP4 route information.

3. Enter the **show ip bgp** command.

```
device# show ip bgp

Total number of BGP Routes: 491790
Status codes: s suppressed, d damped, h history, * valid, > best, i internal, S stale
Origin codes: i - IGP, e - EGP, ? - incomplete
    Network            Next Hop        MED        LocPrf     Weight Path
*>  1.1.1.1/32         0.0.0.0         0          100         32768  ?
*>  2.2.2.2/32         53.0.16.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.17.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.18.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.19.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.20.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.21.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.22.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.23.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.24.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.25.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.26.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.27.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.28.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.29.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.30.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.31.1       0          100         0      200 ?
*   2.2.2.2/32         53.0.8.1        0          100         0      200 ?
…
```

This example shows general BGP4 information.

4. Enter the **show ip bgp attribute-entries** command.

```
device# show ip bgp attribute-entries

Total number of BGP Attribute Entries: 130
1       Next Hop  : 53.0.38.1          MED      :0               Origin:INCOMP
        Originator:0.0.0.0            Cluster List:None
        Aggregator:AS Number :0            Router-ID:0.0.0.0          Atomic:None
        Local Pref:100               Communities:Internet
        AS Path   :200 (length 3)
           AsPathLen: 1  AsNum: 1, SegmentNum: 1, Neighboring As: 200, Source As 200
        Address: 0x645146ec  Hash:255 (0x03000292)
        Links: 0x64514e5c, 0x00000000
        Reference Counts: 66:0:0, Magic: 53
2       Next Hop  : 53.0.39.1          MED      :0               Origin:IGP
        Originator:0.0.0.0            Cluster List:None
        Aggregator:AS Number :0            Router-ID:0.0.0.0          Atomic:None
        Local Pref:100               Communities:Internet
        AS Path   :200 1 (length 5)
           AsPathLen: 2  AsNum: 2, SegmentNum: 1, Neighboring As: 200, Source As 1
        Address: 0x64514e5c  Hash:255 (0x0100039a)
        Links: 0x6451635c, 0x645146ec
        Reference Counts: 7500:0:0, Magic: 70
       …
```

This example shows information about one route-attribute entry that is stored in device memory.

5. Enter the **show ip bgp peer-group** command.

```
device# show ip bgp peer-group

1   BGP peer-group is pg_vrf10, Remote AS: 4001183001
    Description: bgp_vrf4@-4001183001
       MD5 Password: $MiJTfXJVRzMxTTNRUVpaVzhRUVo=
      Address family : IPV4 Unicast
        activate
      Address family : IPV6 Unicast
        no activate
    Currently there are no members.

2   BGP peer-group is pg_vrf11, Remote AS: 4001183011
    Description: bgp_vrf4@-4001183011
       MD5 Password: $MiJTfXJVRzMxTTNRUVpaVzhRWlo=
      Address family : IPV4 Unicast
        activate
      Address family : IPV6 Unicast
        no activate
    Currently there are no members.
```

This example shows output for two BGP peer groups, called “pg” and “pg6”.

6. Enter the **show ip bgp routes** command using the **summary** keyword.

```
device# show ip bgp routes summary

Total number of BGP routes (NLRIs) Installed       : 491790
  Distinct BGP destination networks                  : 15068
  Filtered bgp routes for soft reconfig              : 0
  Routes originated by this router                   : 66
  Routes selected as BEST routes                     : 15068
  Routes Installed as BEST routes                    : 15068
  BEST routes not installed in IP forwarding table   : 0
  Static routes not installed in IP forwarding table : 0
  Unreachable routes (no IGP route for NEXTHOP)      : 0
  IBGP routes selected as best routes                : 0
  EBGP routes selected as best routes                : 15002
  BEST routes not valid for IP forwarding table      : 0
```

This example shows summarized BGP4 route information.

Displaying BGP4 neighbor statistics

Various **show ip bgp neighbor** commands verify information about BGP4 neighbor configurations.

Use one or more of the following commands to verify BGP4 neighbor information. The commands do not have to be entered in this order.

1. Enter the **show ip bgp neighbors** command.

```
device> show ip bgp neighbors

    '+': Data in InQueue '>': Data in OutQueue '-': Clearing
    '*': Update Policy 'c': Group change 'p': Group change Pending
    'r': Restarting 's': Stale '^': Up before Restart '<': EOR waiting

1   IP Address: 60.60.60.20, AS: 200 (IBGP), RouterID: 60.60.60.20, VRF: default-vrf
    State: ESTABLISHED, Time: 4h3m28s, KeepAliveTime: 60, HoldTime: 180
       KeepAliveTimer Expire in 0 seconds, HoldTimer Expire in 159 seconds
    Minimal Route Advertisement Interval: 0 seconds
       RefreshCapability: Received
    Address Family : IPV4 Unicast
       Configured with Add-Path(send receive)capability
       Received Add-Path (send receive)capability in open msg
       Negotiated Add-Path(send receive)capability
    Messages:    Open        Update      KeepAlive   Notification   Refresh-Req
       Sent    : 1           1           275         0              0
       Received: 1           1           275         0              0
    Last Update Time: NLRI       Withdraw          NLRI       Withdraw
                  Tx: 4h3m28s    ---           Rx: 4h3m28s    ---
```

   This example output gives general information about BGP4 neighbors.

2. Enter the **show ip bgp neighbors advertised-routes** command.

```
device> show ip bgp neighbors 192.168.4.211 advertised-routes

       There are 2 routes advertised to neighbor 192.168.4.211
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST I:IBGP L:LOCAL
       Network        Next Hop        Metric  LocPrf   Weight   Status
1      10.102.0.0/24  192.168.2.102   12               32768    BL
2      10.200.1.0/24  192.168.2.102   0                32768    BL
```

   This example shows information about all the routes the BGP4 networking device advertised to the neighbor.

3. Enter the **show ip bgp neighbors** command and specify an IP address.

```
device> show ip bgp neighbors 10.4.0.2

    Total number of BGP neighbors:
1   IP Address: 10.4.0.2, AS: 5 (EBGP), RouterID: 10.0.0.1
       Description: neighbor 10.4.0.2
    Local AS: 101
    State: ESTABLISHED, Time: 0h1m0s, KeepAliveTime: 0, HoldTime: 0
       PeerGroup: pg1
       Multihop-EBGP: yes, ttl: 1
       RouteReflectorClient: yes
       SendCommunity: yes
       NextHopSelf: yes
       DefaultOriginate: yes (default sent)
       MaximumPrefixLimit: 90000
       RemovePrivateAs: : yes
       RefreshCapability: Received
    Route Filter Policies:
       Distribute-list: (out) 20
       Filter-list: (in) 30
       Prefix-list: (in) pf1
       Route-map: (in) setnp1  (out) setnp2
    Messages:    Open    Update  KeepAlive Notification Refresh-Req
       Sent    : 1       1       1         0            0
       Received: 1       8       1         0            0
    Last Update Time: NLRI       Withdraw          NLRI       Withdraw
                 Tx: 0h0m59s    ---           Rx: 0h0m59s    ---
    Last Connection Reset Reason:Unknown
      Notification Sent:     Unspecified
      Notification Received: Unspecified
    TCP Connection state: ESTABLISHED
       Local host:  10.4.0.1, Local  Port: 179
       Remote host: 10.4.0.2, Remote Port: 8053
       ISentSeq:   52837276  SendNext:   52837392  TotUnAck:          0
       TotSent:         116  ReTrans:           0  UnAckSeq:   52837392
       IRcvSeq:  2155052043  RcvNext:  2155052536  SendWnd:       16384
       TotalRcv:        493  DupliRcv:          0  RcvWnd:        16384
       SendQue:           0  RcvQue:            0  CngstWnd:       1460
```

This example shows information about a specifies BGP4 neighbor.

4. Enter the **show ip bgp neighbors received-routes** command.

```
device> show ip bgp neighbor 10.168.4.106 received-routes

       There are 97345 received routes from  neighbor 10.168.4.106
Searching for matching routes, use ^C to quit...
tatus A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH S:SUPPRESSED F:FILTEREDtatus A:AGGREGATE B:BEST
b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH S:SUPPRESSED F:FILTERED
       Prefix             Next Hop        MED     LocPrf    Weight Status
1      10.3.0.0/8          10.168.4.106           100       0      BE
         AS_PATH: 65001 4355 701 8
2      10.4.0.0/8          10.168.4.106           100       0      BE
         AS_PATH: 65001 4355 1
3      10.60.212.0/22      10.168.4.106           100       0      BE
         AS_PATH: 65001 4355 701 1 189
4      10.6.0.0/8          10.168.4.106           100       0      BE
```

This example lists all route information received in route updates from BGP4 neighbors of the device since the soft-reconfiguration feature was enabled.

5. Enter the **show ip bgp neighbors rib-out-routes** command.

```
device> show ip bgp neighbor 192.168.4.211 rib-out-routes 192.168.1.0/24

Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST I:IBGP L:LOCAL
       Prefix             Next Hop        Metric     LocPrf     Weight Status
1      10.200.1.0/24       0.0.0.0         0          101        32768  BL
```

This example shows information about BGP4 outbound RIB routes.

Displaying BGP4 neighbor statistics

Various **show ip bgp neighbor** commands verify information about BGP4 neighbor configurations.

Use one or more of the following commands to verify BGP4 neighbor information. The commands do not have to be entered in this order.

1. Enter the **show ip bgp neighbors** command.

```
device# show ip bgp neighbors

   Total number of BGP Neighbors: 2
1   IP Address: 123.123.123.3, AS: 333 (EBGP), RouterID: 9.9.9.9, VRF: default-vrf
    State: ESTABLISHED, Time: 0h1m32s, KeepAliveTime: 60, HoldTime: 180
       KeepAliveTimer Expire in 17 seconds, HoldTimer Expire in 147 seconds
    Minimal Route Advertisement Interval: 0 seconds
       RefreshCapability: Received
    Messages:    Open    Update  KeepAlive Notification Refresh-Req
       Sent    : 2       15      3339      1            0
       Received: 2       0       3356      0            0
    Last Update Time: NLRI              Withdraw                NLRI                Withdraw
                  Tx: 0h1m32s           ---                 Rx: ---                 ---
    Last Connection Reset Reason:User Reset Peer Session
    Notification Sent:     Cease/Administrative Reset
    Notification Received: Unspecified
    Neighbor NLRI Negotiation:
      Peer configured for IPV4 unicast  Routes
    Neighbor ipv6 MPLS Label Capability Negotiation:
    Neighbor AS4 Capability Negotiation:
    Outbound Policy Group:
       ID: 2, Use Count: 2
    BFD:Disabled
       Byte Sent:   146, Received: 0
       Local host:  123.123.123.2, Local  Port: 44575
       Remote host: 123.123.123.3, Remote Port: 179

2   IP Address: 160.160.160.10, AS: 111 (EBGP), RouterID: 193.24.0.1, VRF: default-vrf
    State: ESTABLISHED, Time: 0h1m33s, KeepAliveTime: 30, HoldTime: 90
       KeepAliveTimer Expire in 12 seconds, HoldTimer Expire in 86 seconds
    Minimal Route Advertisement Interval: 0 seconds
       RefreshCapability: Received
    Messages:    Open    Update  KeepAlive Notification Refresh-Req
       Sent    : 8       0       553       5            0
       Received: 8       9       498       0            0
    Last Update Time: NLRI              Withdraw                NLRI                Withdraw
                  Tx: ---               ---                 Rx: 0h1m33s             ---
    Last Connection Reset Reason:User Reset Peer Session
    Notification Sent:     Cease/Administrative Reset
    Notification Received: Unspecified
    Neighbor NLRI Negotiation:
      Peer configured for IPV4 unicast  Routes
    Neighbor ipv6 MPLS Label Capability Negotiation:
    Neighbor AS4 Capability Negotiation:
    Outbound Policy Group:
       ID: 2, Use Count: 2
    BFD:Disabled
       Byte Sent:   121, Received: 0
       Local host:  160.160.160.20, Local  Port: 53791
       Remote host: 160.160.160.10, Remote Port: 179
```

This example output gives general information about BGP4 neighbors.

2. Enter the **show ip bgp neighbors advertised-routes** command.

```
device# show ip bgp neighbors 123.123.123.3 advertised-routes

       There are 5 routes advertised to neighbor 123.123.123.3
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      110.110.110.0/24   123.123.123.2    0                     0      BE
         AS_PATH: 222 111
2      110.110.111.0/24   123.123.123.2    0                     0      BE
         AS_PATH: 222 111
3      110.110.112.0/24   123.123.123.2    0                     0      BE
         AS_PATH: 222 111
4      110.110.113.0/24   123.123.123.2    0                     0      BE
         AS_PATH: 222 111
5      110.110.114.0/24   123.123.123.2    0                     0      BE
         AS_PATH: 222 111
```

This example shows information about all the routes the BGP4 networking device advertised to the neighbor.

3. Enter the **show ip bgp neighbors last-packet-with-error** command.

```
device# show ip bgp neighbors 123.123.123.3 last-packet-with-error

 Received Message Length: 45
 BGP Message:
  0xffffffff  0xffffffff  0xffffffff  0xffffffff  0x002d0104
  0x014b00b4  0x09090909  0x10020601  0x04010000  0x01020202
  0x00020280  0x00

 BGP Header
  Marker:   0xffffffff  0xffffffff  0xffffffff  0xffffffff
  Message Length: (0x002d) 45
  Message Type: (0x01) OPEN

 OPEN Message
 Version: (0x04) 4
 AS Number: (0x014b) 331
 Hold Time: (0x00b4) 180
 BGP Identifier: (0x09090909) 9.9.9.9
 Optional Parameter length: (0x10) 16

 OPEN message optional parameters
  Parameter Type: (0x02) Capability
  Parameter Length: (0x06) 6
   Capability Type: (0x01) MULTIPROTOCOL EXTENSIONS
   Capability Length: (0x04) 4
   AFI: (0x0100) Unknown(256)
   Reserved: (0x00) 0
   SAFI: (0x01) Unicast

  Parameter Type: (0x02) Capability
  Parameter Length: (0x02) 2
   Capability Type: (0x02) ROUTE REFRESH(new)
   Capability Length: (0x00) 0

  Parameter Type: (0x02) Capability
  Parameter Length: (0x02) 2
   Capability Type: (0x80) ROUTE REFRESH(old)
   Capability Length: (0x00) 0
```

This example shows information about the last packet that contained an error from any of a device's neighbors.

4. Enter the **show ip bgp neighbors received-routes** command.

```
device# show ip bgp neighbors 160.160.160.10 received-routes

       There are 5 received routes from neighbor 160.160.160.10
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      110.110.110.0/24   160.160.160.10   0         100        0      BE
         AS_PATH: 111
2      110.110.111.0/24   160.160.160.10   0         100        0      BE
         AS_PATH: 111
3      110.110.112.0/24   160.160.160.10   0         100        0      BE
         AS_PATH: 111
4      110.110.113.0/24   160.160.160.10   0         100        0      BE
         AS_PATH: 111
5      110.110.114.0/24   160.160.160.10   0         100        0      BE
         AS_PATH: 111
```

This example lists all route information received in route updates from BGP4 neighbors of the device since the soft-reconfiguration feature was enabled.

5. Enter the **show ip bgp neighbors rib-out-routes** command.

```
device# show ip bgp neighbors 123.123.123.3 rib-out-routes

       There are 5 RIB_out routes for neighbor 123.123.123.3
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      110.110.110.0/24   160.160.160.10   0         100        0      BE
         AS_PATH: 111
2      110.110.111.0/24   160.160.160.10   0         100        0      BE
         AS_PATH: 111
3      110.110.112.0/24   160.160.160.10   0         100        0      BE
         AS_PATH: 111
4      110.110.113.0/24   160.160.160.10   0         100        0      BE
         AS_PATH: 111
5      110.110.114.0/24   160.160.160.10   0         100        0      BE
         AS_PATH: 111
```

This example shows information about BGP4 outbound RIB routes.

Displaying BGP4 neighbor statistics

Various **show ip bgp neighbor** commands verify information about BGP4 neighbor configurations.

Use one or more of the following commands to verify BGP4 neighbor information. The commands do not have to be entered in this order.

1. Enter the **show ip bgp neighbors** command.

```
device# show ip bgp neighbors

  Total number of BGP routes (NLRIs) Installed       : 491790
  Distinct BGP destination networks                  : 15068
  Filtered bgp routes for soft reconfig              : 0
  Routes originated by this router                   : 66
  Routes selected as BEST routes                     : 15068
  Routes Installed as BEST routes                    : 15068
  BEST routes not installed in IP forwarding table   : 0
  Static routes not installed in IP forwarding table : 0
  Unreachable routes (no IGP route for NEXTHOP)      : 0
  IBGP routes selected as best routes                : 0
  EBGP routes selected as best routes                : 15002
  BEST routes not valid for IP forwarding table      : 0
F115# show ip bgp neighbors
    Total number of BGP Neighbors: 65
1   IP Address: 52.0.0.2, AS: 2 (EBGP), RouterID: 192.0.137.54, VRF: default-vrf
    State: ESTABLISHED, Time: 6h5m58s, KeepAliveTime: 30, HoldTime: 90
       KeepAliveTimer Expire in 8 seconds, HoldTimer Expire in 63 seconds
    Minimal Route Advertisement Interval: 0 seconds
       GracefulRestartCapability: Received
           Restart Time 90 sec, Restart bit 0
           afi/safi 1/1, Forwarding bit 0
       GracefulRestartCapability: Sent
           Restart Time 120 sec, Restart bit 0
           afi/safi 1/1, Forwarding bit 0
    Messages:    Open    Update  KeepAlive Notification Refresh-Req
       Sent    : 1       28      817       0            0
       Received: 1       9       733       0            0
    Last Update Time: NLRI              Withdraw                NLRI                Withdraw
                  Tx: 6h5m55s           ---                 Rx: 6h5m57s            ---
    Last Connection Reset Reason:Unknown
    Notification Sent:     Unspecified
    Notification Received: Unspecified
    Neighbor NLRI Negotiation:
      Peer Negotiated IPV4  unicast  capability
      Peer configured for IPV4 unicast  Routes
    Neighbor AS4 Capability Negotiation:
    Outbound Policy Group:
       ID: 2, Use Count: 65
       Last update time was 21972 sec ago
    BFD:Disabled
       Byte Sent:   47157, Received: 0
       Local host:  52.0.0.1, Local  Port: 179
       Remote host: 52.0.0.2, Remote Port: 61808

2   IP Address: 53.0.0.1, AS: 200 (EBGP), RouterID: 2.2.2.2, VRF: default-vrf
    State: ESTABLISHED, Time: 6h5m56s, KeepAliveTime: 60, HoldTime: 180
       KeepAliveTimer Expire in 0 seconds, HoldTimer Expire in 125 seconds
    Minimal Route Advertisement Interval: 0 seconds
       RefreshCapability: Received
       GracefulRestartCapability: Received
           Restart Time 120 sec, Restart bit 0
           afi/safi 1/1, Forwarding bit 0
       GracefulRestartCapability: Sent
           Restart Time 120 sec, Restart bit 0
           afi/safi 1/1, Forwarding bit 0
    Messages:    Open    Update  KeepAlive Notification Refresh-Req
       Sent    : 1       46      414       0            0
       Received: 1       45      411       0            0
    Last Update Time: NLRI              Withdraw                NLRI                Withdraw
                  Tx: 6h5m56s           ---                 Rx: 6h5m51s            6h5m55s
    Last Connection Reset Reason:Unknown
    Notification Sent:     Unspecified
    Notification Received: Unspecified
    Neighbor NLRI Negotiation:
```

```
      Peer Negotiated IPV4  unicast  capability
      Peer configured for IPV4 unicast  Routes
    Neighbor AS4 Capability Negotiation:
    Outbound Policy Group:
       ID: 2, Use Count: 65
       Last update time was 21977 sec ago
    BFD:Disabled
       Byte Sent:   70268, Received: 0
       Local host:  53.0.0.2, Local  Port: 8128
       Remote host: 53.0.0.1, Remote Port: 179
```

This example output gives general information about BGP4 neighbors.

2. Enter the **show ip bgp neighbors advertised-routes** command.

```
device# show ip bgp neighbors 52.0.0.2 advertised-routes

       There are 7568 routes advertised to neighbor 52.0.0.2
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      101.0.0.0/24       52.0.0.1         none                  0      BME
         AS_PATH: 100 200 1
2      101.0.1.0/24       52.0.0.1         none                  0      BME
         AS_PATH: 100 200 1
3      101.0.2.0/24       52.0.0.1         none                  0      BME
         AS_PATH: 100 200 1
4      101.0.3.0/24       52.0.0.1         none                  0      BME
         AS_PATH: 100 200 1
5      101.0.4.0/24       52.0.0.1         none                  0      BME
         AS_PATH: 100 200 1
6      101.0.5.0/24       52.0.0.1         none                  0      BME
         AS_PATH: 100 200 1
7      101.0.6.0/24       52.0.0.1         none                  0      BME
         AS_PATH: 100 200 1
8      101.0.7.0/24       52.0.0.1         none                  0      BME
         AS_PATH: 100 200 1
9      101.0.8.0/24       52.0.0.1         none                  0      BME
         AS_PATH: 100 200 1
10     101.0.9.0/24       52.0.0.1         none                  0      BME
         AS_PATH: 100 200 1
 …
```

This example shows information about all the routes the BGP4 networking device advertised to the neighbor.

3. Enter the **show ip bgp neighbors last-packet-with-error** command.

```
device# show ip bgp neighbors 118.113.0.1 last-packet-with-error

 Received Message Length: 19
 BGP Message:
  0xffffffff  0xffffffff  0xffffffff  0xffffffff  0x001304

 BGP Header
  Marker:   0xffffffff  0xffffffff  0xffffffff  0xffffffff
  Message Length: (0x0013) 19
  Message Type: (0x04) KEEPALIVE
```

This example shows information about the last packet that contained an error from any of a device's neighbors.

4. Enter the **show ip bgp neighbors received-routes** command.

```
device# show ip bgp neighbors 118.113.0.1 received-routes

       There are 106 received routes from neighbor 118.113.0.1
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      101.34.0.0/24      118.113.0.1      none      100        0      E
         AS_PATH: 118 2
2      102.34.0.0/24      118.113.0.1      none      100        0      E
         AS_PATH: 118 2
3      112.113.114.0/28   118.113.0.1      none      100        0      E
         AS_PATH: 118 2
4      114.4.1.0/31       118.113.0.1      none      100        0      E
         AS_PATH: 118 2
5      114.114.114.114/32 118.113.0.1      none      100        0      E
         AS_PATH: 118 2
6      118.114.0.0/31     118.113.0.1      none      100        0      E
         AS_PATH: 118 2
7      119.219.32.0/20    118.113.0.1     1          100        0      BE
         AS_PATH: 118
8      119.219.48.0/20    118.113.0.1     1          100        0      BE
         AS_PATH: 118
9      119.219.64.0/20    118.113.0.1     1          100        0      BE
         AS_PATH: 118
10     119.219.80.0/20    118.113.0.1     1          100        0      BE
         AS_PATH: 118
```

This example lists all route information received in route updates from BGP4 neighbors of the device since the soft-reconfiguration feature was enabled.

5. Enter the **show ip bgp neighbors rib-out-routes** command.

```
device# show ip bgp neighbors 53.0.5.1 rib-out-routes
       There are 15068 RIB_out routes for neighbor 53.0.5.1
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      101.0.0.0/24       53.0.16.1        none      100        0      BME
         AS_PATH: 200 1
2      101.0.1.0/24       53.0.16.1        none      100        0      BME
         AS_PATH: 200 1
3      101.0.2.0/24       53.0.16.1        none      100        0      BME
         AS_PATH: 200 1
4      101.0.3.0/24       53.0.16.1        none      100        0      BME
         AS_PATH: 200 1
5      101.0.4.0/24       53.0.16.1        none      100        0      BME
         AS_PATH: 200 1
6      101.0.5.0/24       53.0.16.1        none      100        0      BME
         AS_PATH: 200 1
7      101.0.6.0/24       53.0.16.1        none      100        0      BME
         AS_PATH: 200 1
8      101.0.7.0/24       53.0.16.1        none      100        0      BME
         AS_PATH: 200 1
9      101.0.8.0/24       53.0.16.1        none      100        0      BME
         AS_PATH: 200 1
10     101.0.9.0/24       53.0.16.1        none      100        0      BME
         AS_PATH: 200 1
…
```

This example shows information about BGP4 outbound RIB routes.

Clearing BGP4 dampened paths

BGP4 suppressed routes can be reactivated using a CLI command.

The **show ip bgp dampened-paths** command is entered to verify that there are BGP4 dampened routes. The **clear ip bgp dampening** command is entered to reactivate all suppressed BGP4 routes. The **show ipv6 bgp dampened-paths** command is re-entered to verify that the suppressed BGP4 routes have been reactivated.

1. Enter the **exit** command until you return to Privileged EXEC mode.

```
device(config)# exit
```

2. Enter the **show ip bgp dampened-paths** command to display all BGP4 dampened routes.

```
device# show ip bgp dampened-paths

    Status Code  >:best d:damped h:history *:valid
    Network            From            Flaps Since       Reuse       Path
*d  110.110.114.0/24   160.160.160.10  38    0 :3 :49    0 :10:10    111
*d  110.110.113.0/24   160.160.160.10  38    0 :3 :49    0 :10:10    111
*d  110.110.112.0/24   160.160.160.10  38    0 :3 :49    0 :10:10    111
*d  110.110.111.0/24   160.160.160.10  38    0 :3 :49    0 :10:10    111
*d  110.110.110.0/24   160.160.160.10  38    0 :3 :49    0 :10:10    111
```

3. Enter the **clear ip bgp dampening** command to reactivate all suppressed BGP4 routes.

```
device# clear ip bgp dampening
```

4. Enter the **show ip bgp dampened-paths** command to verify that there are no BGP4 dampened routes.

```
device# show ip bgp dampened-paths
device#
```

The following example reactivates all suppressed BGP4 routes and verifies that there are no suppressed routes.

```
device(config-bgp-router)# exit
device(config)# exit
device# show ip bgp dampened-paths
device# clear ip bgp dampening
device# show ip bgp dampened-paths
```

Clearing BGP4 dampened paths

BGP4 suppressed routes can be reactivated using a CLI command.

The **show ip bgp dampened-paths** command is entered to verify that there are BGP4 dampened routes. The **clear ip bgp dampening** command is entered to reactivate all suppressed BGP4 routes. The **show ipv6 bgp dampened-paths** command is re-entered to verify that the suppressed BGP4 routes have been reactivated.

1. Enter the **exit** command until you return to Privileged EXEC mode.

```
device(config)# exit
```

2. Enter the **show ip bgp dampened-paths** command to display all BGP4 dampened routes.

```
device# show ip bgp dampened-paths

    Status Code  >:best d:damped h:history *:valid
    Network            From            Flp Since        Reuse       Pnlty rIdx dBlk
*d  101.0.99.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.98.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.97.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.96.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.95.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.94.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.93.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.92.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.91.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.90.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.89.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.88.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.87.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.86.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
*d  101.0.85.0/24        12.12.1.2       3   0 :1 :34    0 :29:0     2818  5   0
```

3. Enter the **clear ip bgp dampening** command to reactivate all suppressed BGP4 routes.

```
device# clear ip bgp dampening
```

4. Enter the **show ip bgp dampened-paths** command to verify that there are no BGP4 dampened routes.

```
device# show ip bgp dampened-paths
device#
```

The following example reactivates all suppressed BGP4 routes and verifies that there are no suppressed routes.

```
device(config-bgp-router)# exit
device(config)# exit
device# show ip bgp dampened-paths
device# clear ip bgp dampening
device# show ip bgp dampened-paths
```

BGP PIC

While normal BGP-based convergence can take up minutes (depending on the number of prefixes), BGP Prefix-Independent Convergence (PIC) is an IETF standards-based method that speeds up data path convergence (to subseconds) under failover conditions. PIC is a data-plane feature that does not affect the control plane.

Functional overview

BGP registers with the routing information base manager (RIBM) to resolve a BGP next-hop, which in turn may be reachable through an Interior Gateway Protocol (IGP) prefix. Ultimately, BGP uses the IGP next-hop (instead of the BGP next-hop) to install the prefixes in the RIBM and the forwarding information base (FIB). This scheme worked well for small routing table sizes, and was considered advantageous for data-plane implementations, as the number of lookups required is constant and results in higher throughput. In the context of BGP PIC, such an implementation is loosely referred to as a "flat" RIBM/FIB.

Given the above implementation scenario, assume that because of a change in network topology, the BGP next-hop reachability information changes. In this situation, when the RIBM informs BGP about the change, BGP reruns the best route calculations for the affected prefixes, which could take a while depending on the number of prefixes affected. Also during this period, data traffic continues to get black-holed. In summary, convergence time becomes proportional to the number of prefixes.

As the Internet routing table has grown, and accordingly the number of prefixes in a border router scenario, the flat RIBM/FIB approach makes convergence very slow. Hence, a new scheme is required to handle network outages in a prefix-independent way.

While data plane disruption due to network core failures (node or link) can be tackled with the hierarchical RIBM/FIB approach suggested in the BGP PIC standard, for network edge failures (node or link), BGP PIC requires additional path (ECMP or backup) support from BGP to achieve the same level of performance.

Hierarchical RIB/FIB

BGP-provided prefixes point to next-hops that in turn are reachable through the IGP prefix and are ultimately followed by the IGP next-hop. This hierarchy is maintained in the RIBM/FIB.

Now, in case of a IGP topology change in the network core, apart from the device under consideration or an explicit notification from BFD/BGP regarding the reachability of a BGP next-hop, the RIBM/FIB identifies the affected BGP next-hops and updates the next-hop adjacency information in the hardware without affecting the already programmed prefixes.

In a network, as the number of next-hops is always very limited, in either of the above scenarios the time required to update the data plane with new information will be in subseconds with BGP PIC.

While it is most desired to maintain the above forwarding hierarchy in hardware tables as well, so that topology-change corrective actions can be completed in considerably less time (<50 ms), this approach is not supported by hardware resources in SLX-OS. Instead, the hierarchical mapping is maintained at the RIBM/FIB level only. On notification from IGP (or BGP by means of BFD) of a topology change, the RIBM/FIB issues next-hop updates that in turn change the next-hop hardware table. This design may require a little longer time window for the data plane to converge, but it provides a good trade-off between performance and scale.

BGP additional paths

In most commonly followed designs, BGP provides only the best path to the RIBM for route installation.

If the chosen path is an ECMP path and, during a network change, if one of the next-hops is no longer reachable, with the help of the hierarchical RIBM/FIB the affected next-hop can be easily removed from ECMP at the RIBM/FIB as well as at the hardware level.

However, for scenarios where BGP cannot provide an ECMP next-hop, and if faster data-plane convergence is expected in a prefix-independent manner, BGP PIC recommends that BGP provide primary as well as backup paths to the RIB. The backup path is not programmed into hardware, but rather is maintained only at the RIBM/FIB level.

When network changes affect BGP primary next-hop reachability, the RIBM/FIB can identify the affected BGP next-hop and switch to the backup next-hop.

BGP additional paths are supported by BGP PIC, with the following well-known schemes:

- Full mesh iBGP (without a route reflector)
- Add path
- BGP best external

Supported network triggers for failover

The following network triggers are supported and cause PIC to take effect and restore traffic within subseconds.

- BGP notification of triggers:
  - BFD notification of neighbor down
  - Session down due to execution of **neighbor shutdown** on a remote peer
  - Execution of **clear ip bgp neighbor** on a remote peer

  - Configuration removal on a remote peer
- IGP notification of a topology change
- Interface state going down

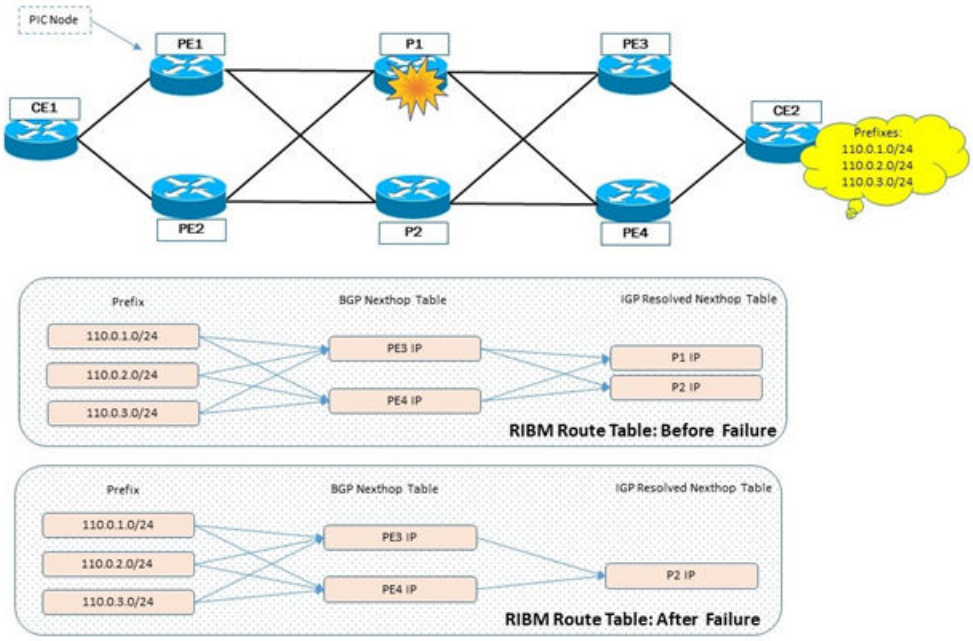
BGP PIC functional details

This section illustrates BGP PIC functionality based on the location and type of network failure.

BGP PIC core node failure

The following figure illustrates a BGP PIC core node failure and the behavior of the route tables.

FIGURE 44 BGP PIC core node failure



The behavior is as follows:

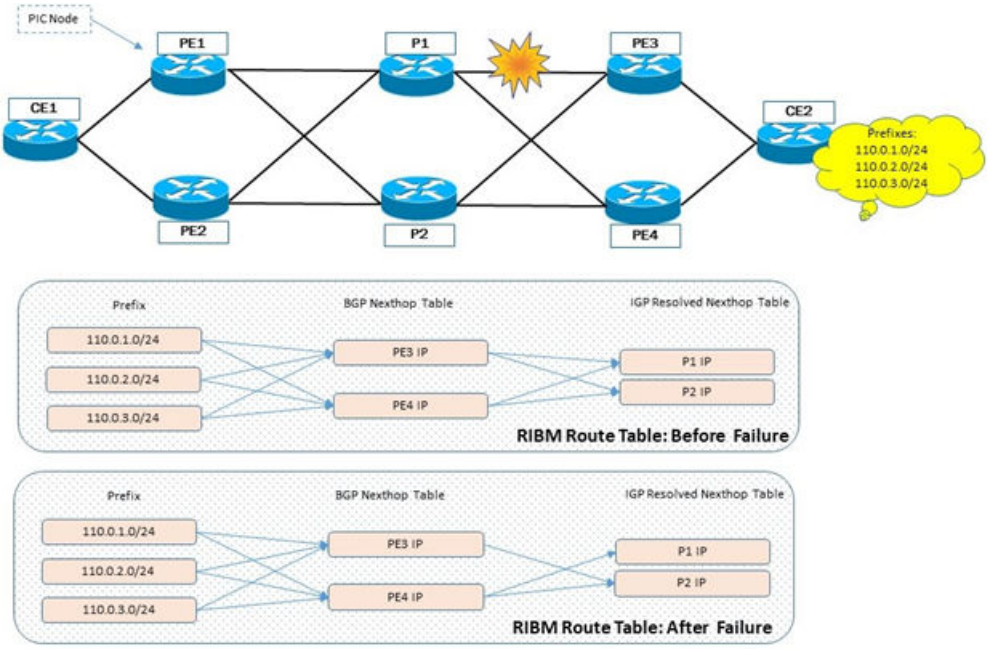
1. At PE1, there are two paths to each of the CE2 prefixes, through PE3 and PE4.
2. BGP next-hop PE3 is reachable through P1 and P2. The same applies to next-hop PE4.
3. When P1 fails, IGP invalidates the path to PE3 through P1; PE3 is now reachable through P2 only.
4. The RIBM/FIB do a reverse lookup and find out which BGP next-hop is affected. Because the number of BGP next-hops is usually less than the number of prefixes, the lookup time is considerably reduced.
5. Once the affected BGP next-hop(s) are identified, the RIBM/FIB issue a next-hop update to the hardware SDK to use the updated list of IGP-resolved next-hops.

In the above example, for both BGP next-hops PE3 and PE4, a next-hop update is issued.

BGP PIC core link failure

The following figure illustrates a BGP PIC core link failure and the behavior of the route tables.

FIGURE 45 BGP PIC core node failure



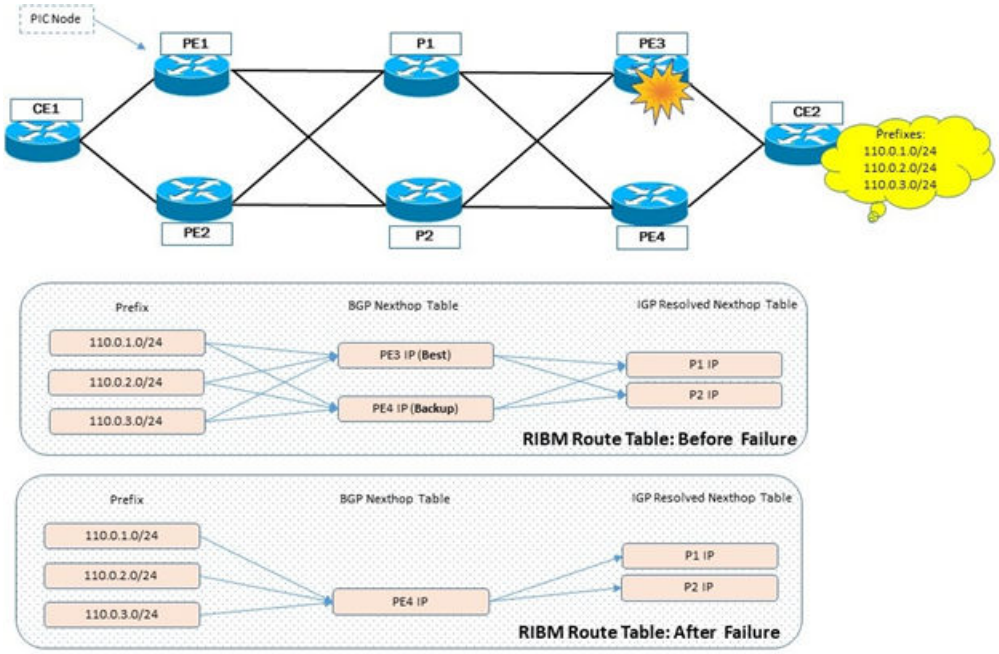
The behavior is as follows:

1. At PE1, there are two paths to each of the CE2 prefixes, through PE3 and PE4.
2. BGP next-hop PE3 is reachable through P1 and P2. The same applies to next-hop PE4.
3. Following a P1–PE3 link failure, IGP invalidates the path to PE3 through P1; PE3 is now reachable through P2 only.
4. The RIBM/FIB do a reverse lookup and find out which BGP next-hop is affected. Because the number of BGP next-hops is usually less than the number of prefixes, the lookup time is considerably reduced.
5. Once the affected BGP next-hop(s) are identified, the RIBM/FIB issue a next-hop update to the hardware SDK to use the updated list of IGP-resolved next-hops.
6. PE3 is now reachable only through P2. Consequently, the RIBM/FIB at PE1 send a next-hop update for PE3 to the hardware SDK with the updated IGP-resolved next-hop (P2).
7. BGP next-hop PE4 continues to be reachable through both P1 and P2.

BGP PIC edge node failure

The following figure illustrates a BGP PIC edge node failure and the behavior of the route tables.

FIGURE 46 BGP PIC edge node failure



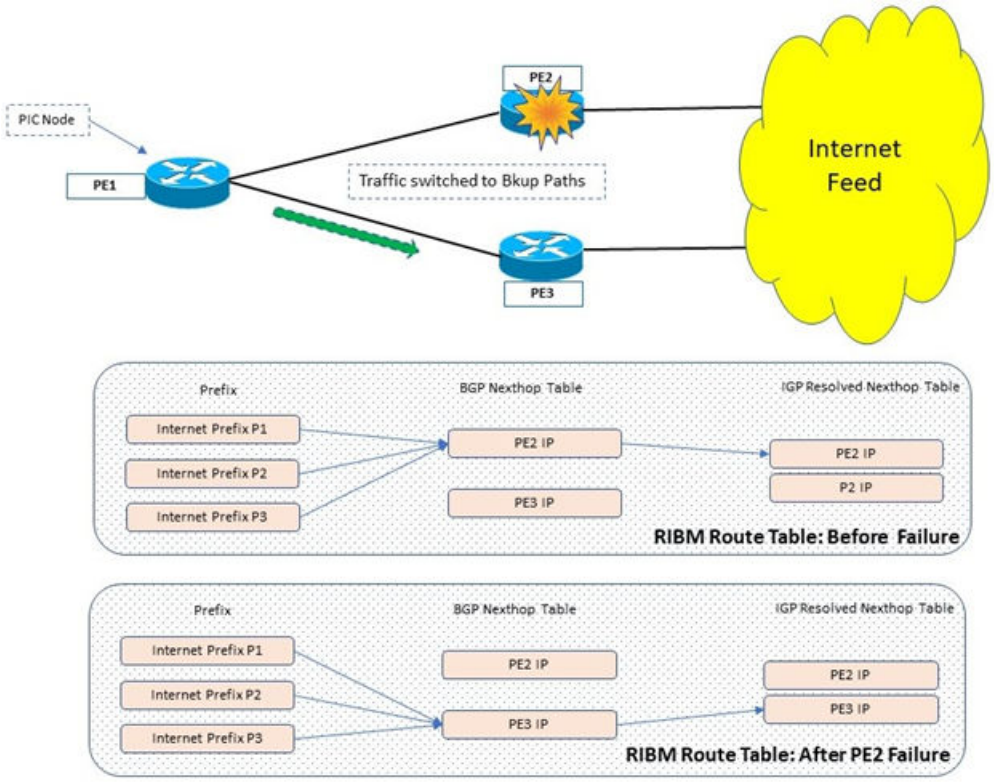
The behavior is as follows:

1. In this scenario, BGP does not form ECMP and instead provide the backup next-hop (PE4) and the primary next-hop (PE3). Note that only primary next-hop is programmed in the hardware SDK.
2. When PE3 fails, the RIBM/FIB at PE1 are notified by BGP that PE3 is no longer reachable.
3. The RIBM/FIB issue a next-hop update to the switch from PE3 to PE4 to the hardware SDK.
4. Now traffic flows through PE4.

BGP with eBGP primary-to-backup failure

The following figure illustrates a BGP with eBGP primary-to-backup failure and the behavior of the route tables.

FIGURE 47 BGP with eBGP primary-to-backup failure



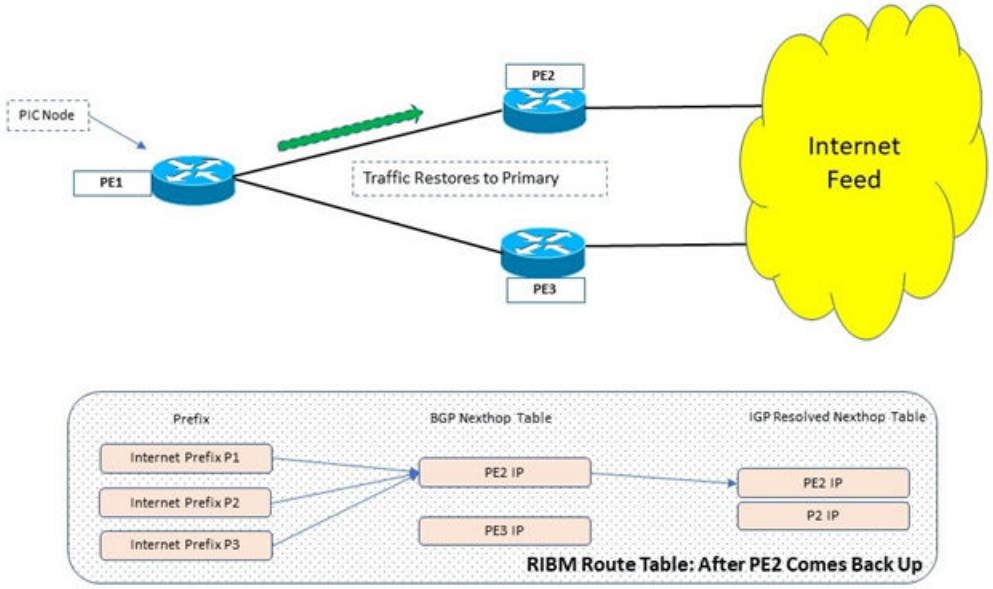
The behavior is as follows:

1. In this scenario BGP does not form ECMP and instead provides backup next-hop PE3 and primary next-hop PE2. Note that only the primary next-hop is programmed in the hardware SDK.
2. When PE2 fails, the RIBM/FIB at PE1 are notified by BGP and conclude that PE3 is no longer reachable.
3. The RIBM/FIB issue a next-hop update to the switch from PE2 to PE3 to the hardware SDK.
4. Now traffic flows through PE3.

BGP with eBGP backup-to-primary failure

The following figure illustrates a BGP with eBGP backup-to-primary failure and the behavior of the route tables.

FIGURE 48 BGP with eBGP backup-to-primary failure



The behavior is as follows:

1. After a while, if PE2 comes back alive and gives the same set of routes to add to the RIBM, the primary path is treated as restored and traffic is switched to the primary again.

BGP PIC considerations and limitations

This section notes the following considerations and limitations for this feature.

Note the following:

- BGP PIC is supported only for IPv4/IPv6 address families. It is not supported for L3VPN/EVPN/IPoMPLS.
- This feature is currently disabled by default.
- Once enabled, this feature is applicable across all VRFs and supported address families.
- Both PIC core and PIC edge use cases are supported.
- PIC is supported on the SLX 9850, the SLX 9640, and the SLX 9540.
- PIC is supported along with high availability. BGP graceful restart must be configured for PIC HA to work.
- This feature is compatible with Optiscale profiles.

Configuring BGP PIC

Do the following to configure BGP Prefix-Independent Convergence.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **prefix-independent-convergence** command.

```
device(config)# prefix-independent-convergence
```

3. Enter the **no** form of the command to disable BGP PIC.

```
device(config)# no prefix-independent-convergence
```

Use the **show running-config** command to confirm the configuration. Backup paths are not displayed in the output.

4. Use the **show ip route** command to confirm changes in the next-hop path.

```
device# show ip route
IP Routing Table for VRF "default-vrf"
Total number of IP routes: 2
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

10.20.30.0/24,
    *via DIRECT, Null0, [1/1], 0m11s, static, tag 0
10.20.31.0/24,
    *via DIRECT, Null0, [1/1], 0m5s, static, tag 0
```

Although the route remains untouched in hardware, the timestamps (highlighted above) reflect changes in the next-hop path.

BGP4+

- BGP4+ overview....383
- BGP global mode....384
- IPv6 unicast address family....387
- BGP4+ neighbors....390
- BGP4+ peer groups....393
- Enabling ASN capability globally....397
- Enabling ASN capability for a BGP4+ neighbor....397
- Importing routes into BGP4+....398
- Advertising the default BGP4+ route....399
- Advertising the default BGP4+ route to a specific neighbor....399
- Using the IPv6 default route as a valid next hop for a BGP4+ route....400
- BGP4+ next hop recursion....401
- BGP4+ NLRIs and next hop attributes....402
- BGP4+ route reflection....403
- BGP4+ route aggregation....405
- BGP4+ multipath....406
- Route maps....408
- Redistributing prefixes into BGP4+....410
- Redistributing routes into BGP4+....411
- Specifying the weight added to BGP4+ received routes....411
- Enabling BGP4+ in a non-default VRF....412
- BGP4+ outbound route filtering....413
- BGP4+ confederations....415
- BGP4+ extended community....416
- BGP4+ graceful restart....420
- Auto shutdown of BGP neighbors on initial configuration....424
- Generalized TTL Security Mechanism support....426
- Disabling the BGP AS_PATH check function....428
- Displaying BGP4+ statistics....429
- Displaying BGP4+ statistics....430
- Displaying BGP4+ neighbor statistics....432
- Displaying BGP4+ neighbor statistics....435
- Clearing BGP4+ dampened paths....437
- Clearing BGP4+ dampened paths....437

BGP4+ overview

The implementation of IPv6 supports multiprotocol BGP (MBGP) extensions that allow Border Gateway Protocol version 4 plus (BGP4+) to distribute routing information. BGP4+ supports all of the same features and functionality as IPv4 BGP (BGP4).

IPv6 MBGP enhancements include:

- An IPv6 unicast address family and network layer reachability information (NLRI)
- An IPv6 multicast address family
- Next hop attributes that use IPv6 addresses

NOTE
The implementation of BGP4+ supports the advertising of routes among different address families. However, it supports BGP4+ unicast routes only; it does not currently support BGP4+ multicast routes.

The implementation of BGP4+ supports the advertising of routes among different address families. BGP4+ multicast and unicast routes are supported. IPv6 unicast-routing is enabled by default.

BGP global mode

Configurations that are not specific to address family configuration are available in the BGP global configuration mode.

```
device(config-bgp-router)# ?

Possible completions:

  address-family                 Enter Address Family command mode
  always-compare-med             Allow comparing MED from different neighbors
  as-path-ignore                 Ignore AS_PATH length for best route selection
  capability                     Set capability
  cluster-id                     Configure Route-Reflector Cluster-ID
  compare-med-empty-aspath    Allow comparing MED from different neighbors even with empty as-path attribute
  compare-routerid               Compare router-id for identical BGP paths
  confederation               Configure AS confederation parameters
  default-local-preference       Configure default local preference value
  distance                       Define an administrative distance
  enforce-first-as               Enforce the first AS for EBGP routes
  fast-external-fallover         Reset session if link to EBGP peer goes down
  install-igp-cost               Install igp cost to nexthop instead of MED value as BGP cost
  local-as                       Configure local AS number
  log-dampening-debug            Log dampening debug messages
  maxas-limit                    Impose limit on number of ASes in AS-PATH attribute
  med-missing-as-worst           Consider routes missing MED attribute as least desirable
  neighbor                       Specify a neighbor router
  timers                         Adjust routing timers

device(config-bgp-router)# ?

Possible completions:
  address-family                 Enter Address Family command mode
  always-compare-med             Allow comparing MED from different neighbors
  as-path-ignore                 Ignore AS_PATH length for best route selection
  auto-shutdown-new-neighbors    Auto shutdown new neighbor
  capability                     Set capability
  cluster-id                     Configure Route-Reflector Cluster-ID
  compare-med-empty-aspath       Allow comparing MED from different neighbors
                                 even with empty as-path attribute
  compare-routerid               Compare router-id for identical BGP paths
  confederation                  Configure AS confederation parameters
  default-local-preference       Configure default local preference value
  describe                       Display transparent command information
  distance                       Define an administrative distance
  enforce-first-as               Enforce the first AS for EBGP routes
  fast-external-fallover         Reset session if link to EBGP peer goes down
  install-igp-cost               Install igp cost to nexthop instead of MED
                                 value as BGP route cost
  local-as                       Configure local AS number
  log-dampening-debug            Log dampening debug messages
  maxas-limit                    Impose limit on number of ASes in AS-PATH
                                 attribute
  med-missing-as-worst           Consider routes missing MED attribute asleast
                                 desirable
  neighbor                       Specify a neighbor router
  timers                         Adjust routing timers

device(config-bgp-router)# ?
```

```
Possible completions:
  address-family                Enter Address Family command mode
  always-compare-med            Allow comparing MED from different neighbors
  as-path-ignore                Ignore AS_PATH length for best route selection
  auto-shutdown-new-neighbors   Auto shutdown new neighbor
  bfd                           Set BFD global parameters for BGP
  capability                    Set capability
  cluster-id                    Configure Route-Reflector Cluster-ID
  compare-med-empty-aspath      Allow comparing MED from different neighbors
                                even with empty as-path attribute
  compare-routerid              Compare router-id for identical BGP paths
  confederation                 Configure AS confederation parameters
  default-local-preference      Configure default local preference value
  describe                      Display transparent command information
  distance                      Define an administrative distance
  enforce-first-as              Enforce the first AS for EBGP routes
  fast-external-fallover        Reset session if link to EBGP peer goes down
  install-igp-cost              Install igp cost to nexthop instead of MED
                                value as BGP route cost
  local-as                      Configure local AS number
  log-dampening-debug           Log dampening debug messages
  maxas-limit                   Impose limit on number of ASes in AS-PATH
                                attribute
  med-missing-as-worst          Consider routes missing MED attribute asleast
                                desirable
  neighbor                      Specify a neighbor router
  timers                        Adjust routing timers

device(config-bgp)# ?

Possible completions:

  address-family                Enter Address Family command mode
  aggregate-address             Configure BGP aggregate entries
  always-compare-med            Allow comparing MED from different neighbors
  always-propagate              Allow readvertisement of best BGP routes not in
                                IP forwarding table
  as-path-ignore                Ignore AS_PATH length info for best route
                                selection
  auto-shutdown-new-neighbors   Automatically set new BGP neighbors to shutdown
                                state
  bfd                           Set BFD global parameters for BGP4
  bfd-enable                    Enable BFD for BGP4
  bgp-redistribute-internal     Allow redistribution of iBGP routes into IGPs
  capability                    Set capability
  clear                         Clear table/statistics/keys
  client-to-client-reflection   Configure client to client route reflection
  cls                           Clear screen
  cluster-id                    Configure Route-Reflector Cluster-ID
  compare-med-empty-aspath      Allow comparing MED from different neighbors
                                even with empty as-path attribute
  compare-routerid              Compare router-id for identical BGP paths
  confederation                 Configure AS confederation parameters
  dampening                     Enable route-flap dampening
  default-information-origina   Distribute a default route
  default-local-preference      Configure default local preference value
  default-metric                Set metric of redistributed routes
  distance                      Define an administrative distance
  enforce-first-as              Enforce the first AS for EBGP routes
  fast-external-fallover        Reset session if link to EBGP peer goes down
  graceful-restart              Enables the BGP graceful restart capability
  install-igp-cost              Install igp cost to nexthop instead of MED
                                value as BGP route cost
  local-as                      Configure local AS number
  log-dampening-debug           Log dampening debug messages
  maxas-limit                   Impose limit on number of ASes in AS-PATH
                                attribute
  maximum-paths                 Forward packets over multiple paths
  med-missing-as-worst          Consider routes missing MED attribute as least
                                desirable
  multipath                     Enable multipath for ibgp or ebgp neighbors only
  neighbor                      Specify a neighbor router
```

```
  network                       Specify a network to announce via BGP
  next-hop-enable-default       Enable default route for BGP next-hop lookup
  next-hop-mpls                 Include MPLS routes for BGP next-hop lookup
  next-hop-recursion            Perform next-hop recursive lookup for BGP route
  redistribute                  Redistribute information from another routing
                                protocol
  rib-route-limit               Limit BGP rib count in routing table
  static-network                Special network that do not depends on IGP and
                                always treat as best route in BGP
  table-map                     Map external entry attributes into routing table
  timers                        Adjust routing timers
  update-time                   Configure igp route update interval
```

The following neighbor configuration options are allowed under BGP global configuration mode:

```
device(config-bgp-router)# neighbor 2001:2018:8192::125 ?

Possible completions:
  advertisement-interval   Minimum interval between sending BGP routing updates
  as-override              Override matching AS-number while sending update
  bfd                      Enable BFD session for the neighbor
  capability               Advertise capability to the peer
  description              Neighbor by description
  ebgp-btsh                Enable EBGP TTL Security Hack Protection
  ebgp-multihop            Allow EBGP neighbors not on directly connected
                           networks
  enforce-first-as         Enforce the first AS for EBGP routes
  local-as                 Assign local-as number to neighbor
  maxas-limit              Impose limit on number of ASes in AS-PATH attribute
  next-hop-self            Disable the next hop calculation for this neighbor
  password                 Enable TCP-MD5 password protection
  peer-group               Create Peer Group
  remote-as                Specify a BGP neighbor
  remove-private-as        Remove private AS number from outbound updates
  shutdown                 Administratively shut down this neighbor
  soft-reconfiguration     Per neighbor soft reconfiguration
  static-network-edge      Neighbor as special service edge, static-network
                           shall not be advertised if installed as DROP
  timers                   BGP per neighbor timers
  update-source            Source of routing updates

device(config-bgp-router)# neighbor 2001:2018:8192::125 ?

Possible completions:
  advertisement-interval   Minimum interval between sending BGP routing updates
  as-override              Override matching AS-number while sending update
  capability               Advertise capability to the peer
  description              Neighbor by description
  ebgp-btsh                Enable EBGP TTL Security Hack Protection
  ebgp-multihop            Allow EBGP neighbors not on directly connected
                           networks
  enforce-first-as         Enforce the first AS for EBGP routes
  local-as                 Assign local-as number to neighbor
  maxas-limit              Impose limit on number of ASes in AS-PATH attribute
  next-hop-self            Disable the next hop calculation for this neighbor
  password                 Enable TCP-MD5 password protection
  peer-group               Create Peer Group
  remote-as                Specify a BGP neighbor
  remove-private-as        Remove private AS number from outbound updates
  shutdown                 Administratively shut down this neighbor
  soft-reconfiguration     Per neighbor soft reconfiguration
  static-network-edge      Neighbor as special service edge, static-network
                           shall not be advertised if installed as DROP
  timers                   BGP per neighbor timers
  update-source            Source of routing updates

device(config-bgp-router)# neighbor 2001:2018:8192::125 ?

Possible completions:
  advertisement-interval   Minimum interval between sending BGP routing updates
```

```
  as-override              Override matching AS-number while sending update
  bfd                      Enable BFD session for the neighbor
  capability               Advertise capability to the peer
  description              Neighbor by description
  ebgp-btsh                Enable EBGP TTL Security Hack Protection
  ebgp-multihop            Allow EBGP neighbors not on directly connected
                           networks
  enforce-first-as         Enforce the first AS for EBGP routes
  local-as                 Assign local-as number to neighbor
  maxas-limit              Impose limit on number of ASes in AS-PATH attribute
  next-hop-self            Disable the next hop calculation for this neighbor
  password                 Enable TCP-MD5 password protection
  peer-group               Create Peer Group
  remote-as                Specify a BGP neighbor
  remove-private-as        Remove private AS number from outbound updates
  shutdown                 Administratively shut down this neighbor
  soft-reconfiguration     Per neighbor soft reconfiguration
  static-network-edge      Neighbor as special service edge, static-network
                           shall not be advertised if installed as DROP
  timers                   BGP per neighbor timers
  update-source            Source of routing updates

device(config-bgp)# neighbor 2001:2018:8192::125 ?

Possible completions:
   activate                 Allow exchange of route in the current family mode
  advertisement-interval   Minimum interval between sending BGP routing updates
  bfd                      Set BFD parameters for BGP4 neighbor
  capability               Advertise capability to the peer
  description              Neighbor by description
  ebgp-btsh                Enable EBGP TTL Security Hack Protection
  ebgp-multihop            Allow EBGP neighbors not on directly connected
                           networks
  enforce-first-as         Enforce the first AS for EBGP routes
  fail-over                Set Failover source for BGP4 for this peer
  local-as                 Assign local-as number to neighbor
  maxas-limit              Impose limit on number of ASes in AS-PATH attribute
  next-hop-self            Disable the next hop calculation for this neighbor
  password                 Enable TCP-MD5 password protection
  peer-group               Assign peer-group to neighbor
  remote-as                Specify a BGP neighbor
  remove-private-as        Remove private AS number from outbound updates
  shutdown                 Administratively shut down this neighbor
  soft-reconfiguration     Per neighbor soft reconfiguration
  static-network-edge      Neighbor as special service edge, static-network
                           shall not be advertised if installed as DROP
  timers                   BGP per neighbor timers
  update-source            Source of routing updates
```

IPv6 unicast address family

The IPv6 unicast address family configuration level provides access to commands that allow you to configure BGP4+ unicast routes. The commands that you enter at this level apply only to the IPv6 unicast address family.

BGP4+ supports the IPv6 address family configuration level.

You can generate a configuration for BGP4+ unicast routes that is separate and distinct from configurations for IPv4 unicast routes.

You can generate a configuration for BGP4+ unicast routes that is separate and distinct from configurations for IPv4 unicast routes and BGP multicast routes.

The commands that you can access while at the IPv6 unicast address family configuration level are also available at the IPv4 unicast address family configuration levels. Each address family configuration level allows you to access commands that apply to that particular address family only.

Where relevant, this chapter discusses and provides IPv6-unicast-specific examples. You must first configure IPv6 unicast routing for any IPv6 routing protocol to be active.

Where relevant, this chapter discusses and provides IPv6-unicast-specific examples.

The following configuration options are allowed under BGP IPv6 address family unicast mode:

```
device(config-bgp-ipv6u)# ?

Possible completions:

  aggregate-address               Configure BGP aggregate entries
  always-propagate                Allow readvertisement of best BGP routes not
                                  in IP Forwarding table
  bgp-redistribute-internal       Allow redistribution of iBGP routes into IGPs
  client-to-client-reflection     Configure client to client route reflection
  dampening                       Enable route-flap dampening
  default-information-originate   Originate Default Information
  default-metric                  Set metric of redistributed routes
  graceful-restart                Enables the BGP graceful restart capability
  maximum-paths                   Forward packets over multiple paths
  multipath                       Enable multipath for ibgp or ebgp neighbors
                                  only
  neighbor                        Specify a neighbor router
  network                         Specify a network to announce via BGP
  next-hop-enable-default         Enable default route for BGP next-hop lookup
  next-hop-recursion              Perform next-hop recursive lookup for BGP
                                  route
  redistribute                    Redistribute information from another
                                  routing protocol
  rib-route-limit                 Limit BGP rib count in routing table
  table-map                       Map external entry attributes into routing
                                  table
  update-time                     Configure igp route update interval
```

```
device(config-bgp-ipv6u)# ?

Possible completions:

  aggregate-address               Configure BGP aggregate entries
  always-propagate                Allow readvertisement of best BGP routes not
                                  in IP Forwarding table
  bgp-redistribute-internal       Allow redistribution of iBGP routes into IGPs
  client-to-client-reflection     Configure client to client route reflection
  dampening                       Enable route-flap dampening
  default-information-originate   Originate Default Information
  default-metric                  Set metric of redistributed routes
  graceful-restart                Enables the BGP graceful restart capability
  maximum-paths                   Forward packets over multiple paths
  multipath                       Enable multipath for ibgp or ebgp neighbors
                                  only
  neighbor                        Specify a neighbor router
  network                         Specify a network to announce via BGP
  next-hop-enable-default         Enable default route for BGP next-hop lookup
  next-hop-recursion              Perform next-hop recursive lookup for BGP
                                  route
  redistribute                    Redistribute information from another
                                  routing protocol
  rib-route-limit                 Limit BGP rib count in routing table
  table-map                       Map external entry attributes into routing
                                  table
  update-time                     Configure igp route update interval
```

```
device(config-bgp-ipv6u)# ?

Possible completions:

  aggregate-address             Configure BGP aggregate entries
  always-propagate              Allow readvertisement of best BGP routes not in
                                IP forwarding table
  bfd                           Set BFD global parameters for BGP4
```

```
  bfd-enable                    Enable BFD for BGP4
  bgp-redistribute-internal     Allow redistribution of iBGP routes into IGPs
  client-to-client-reflection   Configure client to client route reflection
  dampening                     Enable route-flap dampening
  default-information-origina   Distribute a default route
  default-metric                Set metric of redistributed routes
  graceful-restart              Enables the BGP graceful restart capability
  log-dampening-debug           Log dampening debug messages
  maximum-paths                 Forward packets over multiple paths
  multipath                     Enable multipath for ibgp or ebgp neighbors only
  neighbor                      Specify a neighbor router
  network                       Specify a network to announce via BGP
  next-hop-enable-default       Enable default route for BGP next-hop lookup
  next-hop-recursion            Perform next-hop recursive lookup for BGP route
  redistribute                  Redistribute information from another routing
                                protocol
  rib-route-limit               Limit BGP rib count in routing table
  table-map                     Map external entry attributes into routing table
  update-time                   Configure igp route update interval
```

The following neighbor configuration options are allowed under BGP IPv6 address family unicast mode:

```
device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 ?

Possible completions:
  activate                  Allow exchange of route in the current family mode
  additional-paths          Specify bgp additional paths
  advertisement-interval    Minimum interval between sending BGP routing updates
  allowas-in                Accept as-path with my AS present in it
  as-override               Override matching AS-number while sending update
  bfd                       Set BFD parameters for BGP4 neighbor
  capability                Advertise capability to the peer
  default-originate         Originate default route to peer
  description               Neighbor by description
  ebgp-btsh                 Enable EBGP TTL Security Hack Protection
  ebgp-multihop             Allow EBGP neighbors not on directly connected
                            networks
  fail-over                 Set Failover source for BGP4 for this peer
  filter-list               Establish BGP filters
  local-as                  Assign local-as number to neighbor
  maximum-prefix            Maximum number of prefix accept from this peer
  next-hop-self             Disable the next hop calculation for this neighbor
  password                  Enable TCP-MD5 password protection
  peer-group                Assign peer-group to neighbor
  prefix-list               Prefix List for filtering routes
  remote-as                 Specify a BGP neighbor
  remove-private-as         Remove private AS number from outbound updates
  route-map                 Apply route map to neighbor
  route-reflector-client    Configure a neighbor as Route Reflector client
  send-community            Send community attribute to this neighbor
  shutdown                  Administratively shut down this neighbor
  soft-reconfiguration      Per neighbor soft reconfiguration
  timers                    BGP per neighbor timers
  unsuppress-map            Route-map to selectively unsuppress suppressed routes
  update-source             Source of routing updates
  weight                    Set default weight for routes from this neighbor

device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 ?

Possible completions:
  activate                  Allow exchange of route in the current family mode
  additional-paths          Advertise the additional paths
  allowas-in                Disables the AS_PATH check of the routes learned
                            from the AS
  capability                Advertise capability to the peer
  default-originate         Originate default route to peer
  enable-peer-as-check      Disable routes advertise between peers in same AS
  filter-list               Establish BGP filters
  maximum-prefix            Maximum number of prefix accept from this peer
  prefix-list               Prefix List for filtering routes
  route-map                 Apply route map to neighbor
```

```
  route-reflector-client   Configure a neighbor as Route Reflector client
  send-community           Send community attribute to this neighbor
  unsuppress-map           Route-map to selectively unsuppress suppressed
                           routes
  weight                   Set default weight for routes from this neighbor

device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 ?

Possible completions:
  activate                 Allow exchange of route in the current family mode
  allowas-in               Disables the AS_PATH check of the routes learned
                           from the AS
  capability               Advertise capability to the peer
  default-originate        Originate default route to peer
  filter-list              Establish BGP filters
  maximum-prefix           Maximum number of prefix accept from this peer
  prefix-list              Prefix List for filtering routes
  route-map                Apply route map to neighbor
  route-reflector-client   Configure a neighbor as Route Reflector client
  send-community           Send community attribute to this neighbor
  unsuppress-map           Route-map to selectively unsuppress suppressed
                           routes
  weight                   Set default weight for routes from this neighbor
```

BGP4+ neighbors

BGP4+ neighbors can be configured using link-local addresses or global addresses.

BGP4+ neighbors can be created using link-local addresses for peers in the same link. For link-local peers, the neighbor interface over which the neighbor and local device exchange prefixes is specified through the **neighbor update-source** command, and a route map is configured to set up a global next hop for packets destined for the neighbor.

To configure BGP4+ neighbors that use link-local addresses, you must do the following:

- Add the IPv6 address of a neighbor in a remote autonomous system (AS) to the BGP4+ neighbor table of the local device.
- Identify the neighbor interface over which the neighbor and local device will exchange prefixes using the **neighbor update-source** command.
- Configure a route map to set up a global next hop for packets destined for the neighbor.

The neighbor should be activated in the IPv6 address family configuration mode using the **neighbor activate** command.

BGP4+ neighbors can also be configured using a global address. The global IPv6 address of a neighbor in a remote AS must be added, and the neighbor should be activated in the IPv6 address family configuration mode using the **neighbor activate** command.

Configuring BGP4+ neighbors using global IPv6 addresses

BGP4+ neighbors can be configured using global IPv6 addresses.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

```
device(config-bgp)# local-as 1000
```

5. Enter the **neighbor remote-as** command, specifying an IPv6 address, to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 remote-as 1001
```

```
device(config-bgp)# neighbor 2001:db8:93e8:cc00::1 remote-as 1001
```

6. Enter the **address family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

```
device(config-bgp)# address-family ipv6 unicast
```

7. Enter the **neighbor activate** command to enable the exchange of information with the neighbor.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:93e8:cc00::1 activate
```

The following example configures a neighbor using a global IPv6 address.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 remote-as 1001
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:93e8:cc00::1 activate

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 remote-as 1001
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:93e8:cc00::1 activate

device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# neighbor 2001:db8:93e8:cc00::1 remote-as 1001
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:93e8:cc00::1 activate
```

Configuring BGP4+ neighbors using link-local addresses

BGP4+ neighbors can be configured using link-local addresses.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
device(config-bgp)# local-as 1000
```

4. Enter the **neighbor remote-as** command, specifying an IPv6 address, to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
```

5. Enter the **neighbor update-source** command to specify an interface.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 update-source ethernet 3/1
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 update-source ethernet 0/1
device(config-bgp)# neighbor fe80:4398:ab30:45de::1 update-source ethernet 3/1
```

6. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp)# address-family ipv6 unicast
```

7. Enter the **neighbor activate** command to enable the exchange of information with the neighbor.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
```

8. Enter the **neighbor route-map** command and specify the **out** keyword to apply a route map to outgoing routes.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out myroutemap
```

9. Enter the **exit** command until you return to global configuration mode.

```
device(config-bgp-ipv6u)# exit
```

10. Enter the **route-map** *name* **permit** command to define the route map and enter route map configuration mode.

```
device(config)# route-map myroutemap permit 10
```

11. Enter the **set ipv6 next-hop** command and specify an IPv6 address to set the IPv6 address of the next hop.

```
device(config-routemap-myroutemap)# set ipv6 next-hop 2001::10
device(config-routemapmap-myroutemap/permit/10)# set ipv6 next-hop 2001::10
```

The following example configures a neighbor using a link-local address and configures a route map to set up a global next hop for packets destined for the neighbor.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp)# neighbor fe80:4398:ab30:45de::1 update-source ethernet 3/1
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out myroutemap
device(config-bgp-ipv6u)# exit
device(config)# route-map myroutemap permit 10
device(config-routemap-myroutemap)# set ipv6 next-hop 2001::10

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 update-source ethernet 3/1
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out myroutemap
device(config-bgp-ipv6u)# exit
device(config)# route-map myroutemap permit 10
device(config-routemapmap-myroutemap/permit/10)# set ipv6 next-hop 2001::10

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 update-source ethernet 0/1
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out myroutemap
device(config-bgp-ipv6u)# exit
device(config)# route-map myroutemap permit 10
device(config-routemapmap-myroutemap/permit/10)# set ipv6 next-hop 2001::10
```

BGP4+ peer groups

Neighbors having the same attributes and parameters can be grouped together by means of the **neighbor peer-group** command.

You must first create a peer group, after which you can associate neighbor IPv6 addresses with the peer group. All of the attributes that are allowed on a neighbor are allowed on a peer group as well.

BGP4+ peers and peer groups are activated in the IPv6 address family configuration mode to establish the BGP4+ peering sessions.

An attribute value configured explicitly for a neighbor takes precedence over the attribute value configured on the peer group. In the case where neither the peer group nor the individual neighbor has the attribute configured, the default value for the attribute is used.

> NOTE
> BGP4 neighbors are established and the prefixes are advertised using the **neighbor** *IP address* **remote-as** command in router BGP mode. However, when establishing BGP4+ peer sessions and exchanging IPv6 prefixes, neighbors must also be activated using the **neighbor** *IPv6 address* **activate** command in IPv6 address family configuration mode.

Configuring BGP4+ peer groups

A peer group can be created and neighbor IPv6 addresses can be associated with the peer group. The peer group is then activated in the IPv6 address family configuration mode.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
device(config)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
device(config-bgp)# local-as 1000
```

5. Enter the **neighbor peer-group** command to create a peer group.

```
device(config-bgp-router)# neighbor mypeergroup1 peer-group
device(config-bgp)# neighbor mypeergroup1 peer-group
```

6. Enter the **neighbor remote-as** command, specifying a peer group, to specify the ASN of the peer group.

```
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
device(config-bgp)# neighbor mypeergroup1 remote-as 11
```

7. Enter the **neighbor peer-group** command to associate a neighbor with the peer group.

```
device(config-bgp-router)# neighbor 2001:2018:8192::125 peer-group mypeergroup1
device(config-bgp)# neighbor 2001:2018:8192::125 peer-group mypeergroup1
```

8. Enter the **neighbor peer-group** command to associate another neighbor with the peer group.

```
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group mypeergroup1
device(config-bgp)# neighbor 2001:2018:8192::124 peer-group mypeergroup1
```

9. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp)# address-family ipv6 unicast
```

10. Enter the **neighbor activate** command to establish an IPv6 BGP session with the peer group.

```
device(config-bgp-ipv6u)# neighbor mypeergroup1 activate
```

The following example creates a peer group, specifying two neighbors to belong to the peer group, and activates the peer group.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor mypeergroup1 peer-group
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
device(config-bgp-router)# neighbor 2001:2018:8192::125 peer-group mypeergroup1
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group mypeergroup1
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor mypeergroup1 activate

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor mypeergroup1 peer-group
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
device(config-bgp-router)# neighbor 2001:2018:8192::125 peer-group mypeergroup1
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group mypeergroup1
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor mypeergroup1 activate

device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# neighbor mypeergroup1 peer-group
device(config-bgp)# neighbor mypeergroup1 remote-as 11
device(config-bgp)# neighbor 2001:2018:8192::125 peer-group mypeergroup1
device(config-bgp)# neighbor 2001:2018:8192::124 peer-group mypeergroup1
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor mypeergroup1 activate
```

Configuring a peer group with IPv4 and IPv6 peers

A peer group that contains both IPv4 and IPv6 peers can be configured.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

   ```
   device(config)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

   ```
   device(config-bgp)# local-as 1000
   ```

5. Enter the **neighbor peer-group** command, specifying a peer group, to create a peer group.

   ```
   device(config-bgp-router)# neighbor p1 peer-group
   ```

   ```
   device(config-bgp)# neighbor p1 peer-group
   ```

6. Enter the **neighbor remote-as** command to specify the ASN of the peer group.

```
device(config-bgp-router)# neighbor p1 remote-as 11
device(config-bgp)# neighbor p1 remote-as 11
```

7. Enter the **neighbor peer-group** command, specifying an IPv6 address, to associate a neighbor with the peer group.

```
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group p1
device(config-bgp)# neighbor 2001:2018:8192::124 peer-group p1
```

8. Enter the **neighbor peer-group** command, specifying a different IPv6 address, to associate another neighbor with the peer group.

```
device(config-bgp-router)# neighbor 10.0.0.1 peer-group p1
device(config-bgp)# neighbor 10.0.0.1 peer-group p1
```

9. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp)# address-family ipv6 unicast
```

10. Enter the **neighbor activate** command to establish an IPv6 BGP session with the peer group.

```
device(config-bgp-ipv6u)# neighbor p1 activate
```

The following example creates a peer group with both IPv6 and IPv4 peers and activates the peer group in the IPv6 address family.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor p1 peer-group
device(config-bgp-router)# neighbor p1 remote-as 11
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group p1
device(config-bgp-router)# neighbor 10.0.0.1 peer-group p1
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor p1 activate

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor p1 peer-group
device(config-bgp-router)# neighbor p1 remote-as 11
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group p1
device(config-bgp-router)# neighbor 10.0.0.1 peer-group p1
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor p1 activate

device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# neighbor p1 peer-group
device(config-bgp)# neighbor p1 remote-as 11
device(config-bgp)# neighbor 2001:2018:8192::124 peer-group p1
device(config-bgp)# neighbor 10.0.0.1 peer-group p1
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor p1 activate
```

Enabling ASN capability globally

Four-byte autonomous system numbers (ASNs) can be configured on a device. The following task enables 4-byte autonomous system number (ASN) capability at the BGP global level.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **capability as4-enable** command to enable 4-byte autonomous system number (ASN) capability globally.

   ```
   device(config-bgp-router)# capability as4-enable
   ```

The following example enables 4-byte autonomous system number (ASN) capability globally.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# capability as4-enable
```

Enabling ASN capability for a BGP4+ neighbor

Four-byte autonomous system numbers (ASNs) can be configured for a neighbor or peer-group. The following task enables 4-byte autonomous system number (ASN) capability for a BGP4+ neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

4. Enter the **neighbor capability as4** command with the **enable** parameter, and specify an IPv6 address, to enable 4-byte autonomous system number (ASN) capability for a specific neighbor.

   ```
   device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 capability as4 enable
   ```

The following example enables 4-byte autonomous system number (ASN) capability for a specific BGP4+ neighbor.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 capability as4 enable
```

Importing routes into BGP4+

Routes can be explicitly specified for advertisement by BGP.

The routes imported into BGP4+ must first exist in the IPv6 unicast route table.

The routes imported into BGP4+ must first exist in the IPv6 multicast or unicast route table.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

   ```
   device(config)# router bgp
   ```

4. Enter the **neighbor remote-as** command, specifying an IPv6 address, to specify the ASN in which the remote neighbor resides.

   ```
   device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
   ```

   ```
   device(config-bgp)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
   ```

5. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

   ```
   device(config-bgp)# address-family ipv6 unicast
   ```

6. Enter the **network** command and specify a *network/mask* to import the specified prefix into the BGP4+ database.

   ```
   device(config-bgp-ipv6u)# network 2001:db8::/32
   ```

The following example imports the 2001:db8::/32 prefix in to the BGP4+ database for advertising.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# network 2001:db8::/32

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# network 2001:db8::/32

device# configure terminal
device(config)# router bgp
device(config-bgp)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# network 2001:db8::/32
```

Advertising the default BGP4+ route

A BGP device can be configured to advertise the default IPv6 route to all BGP4+ neighbors and to install that route in the local BGP4+ route table.

The default route must be present in the local IPv6 route table.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

   ```
   device(config)# router bgp
   ```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

   ```
   device(config-bgp)# address-family ipv6 unicast
   ```

5. Enter the **default-information-originate** command to advertise the default IPv6 route to all BGP4+ neighbors and to install that route in the local BGP4+ route table.

   ```
   device(config-bgp-ipv6u)# default-information-originate
   ```

The following example enables a BGP4+ device to advertise the default IPv6 route to all BGP4+ neighbors and to install that route in the local BGP4+ route table.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# default-information-originate

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# default-information-originate

device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# default-information-originate
```

Advertising the default BGP4+ route to a specific neighbor

A BGP device can be configured to advertise the default IPv6 route to a specific neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
device(config)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
device(config-bgp)# local-as 1000
```

5. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp)# address-family ipv6 unicast
```

6. Enter the **neighbor default-originate** command and specify an IPv6 address to enable the BGP4+ device to advertise the default IPv6 route to a specific neighbor.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:93e8:cc00::1 default-originate
```

The following example enables a BGP4+ device to advertise the default IPv6 route to a specific neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:93e8:cc00::1 default-originate

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:93e8:cc00::1 default-originate

device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:93e8:cc00::1 default-originate
```

Using the IPv6 default route as a valid next hop for a BGP4+ route

In certain cases, such as when a device is acting as an edge device, it can be configured to use the default route as a valid next hop.

By default, a device does not use a default route to resolve a BGP4+ next-hop route. If the IPv6 route lookup for the BGP4+ next-hop does not result in a valid IGP route (including static or direct routes), the BGP4+ next-hop is considered to be unreachable and the BGP4+ route is not used. You can configure the device to use the default route as a valid next hop.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

```
device(config-bgp)# address-family ipv6 unicast
```

5. Enter the **next-hop-enable-default** command to configure the device to use the default route as a valid next hop.

```
device(config-bgp-ipv6u)# next-hop-enable-default
```

The following example configures a BGP4+ device to use the default route as a valid next hop.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# next-hop-enable-default
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# next-hop-enable-default
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# next-hop-enable-default
```

BGP4+ next hop recursion

A device can find the IGP route to the next-hop gateway for a BGP4+ route.

For each BGP4+ route learned, the device performs a route lookup to obtain the IPv6 address of the next hop for the route. A BGP4+ route is eligible for addition in the IPv6 route table only if the following conditions are true:

- The lookup succeeds in obtaining a valid next-hop IPv6 address for the route.
- The path to the next-hop IPv6 address is an IGP path or a static route path.

By default, the software performs only one lookup for the next-hop IPv6 address for the BGP4+ route. If the next hop lookup does not result in a valid next hop IPv6 address, or the path to the next hop IPv6 address is a BGP4+ path, the BGP4+ route destination is considered unreachable. The route is not eligible to be added to the IPv6 route table.

The BGP4+ route table can contain a route with a next hop IPv6 address that is not reachable through an IGP route, even though the device can reach a hop farther away through an IGP route. This can occur when the IGPs do not learn a complete set of IGP routes, so the device learns about an internal route through IBGP instead of through an IGP. In this case, the IPv6 route table will not contain a route that can be used to reach the BGP4+ route destination.

To enable the device to find the IGP route to the next-hop gateway for a BGP4+ route, enable recursive next-hop lookups. With this feature enabled, if the first lookup for a BGP4+ route results in an IBGP path that originated within the same AS, rather than an IGP path or static route path, the device performs a lookup on the next hop IPv6 address for the next hop gateway. If this second lookup results in

an IGP path, the software considers the BGP4+ route to be valid and adds it to the IPv6 route table. Otherwise, the device performs another lookup on the next hop IPv6 address of the next hop for the next hop gateway, and so on, until one of the lookups results in an IGP route.

You must configure a static route or use an IGP to learn the route to the EBGP multihop peer.

Enabling next-hop recursion

Next hop recursion can be enabled so that a device can find the IGP route to the next hop gateway for a BGP4+ route.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

   ```
   device(config)# router bgp
   ```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

   ```
   device(config-bgp)# address-family ipv6 unicast
   ```

5. Enter the **next-hop-recursion** command to enable recursive next hop lookups.

   ```
   device(config-bgp-ipv6u)# next-hop-recursion
   ```

The following example enables recursive next hop lookups.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# next-hop-recursion

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# next-hop-recursion

device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# next-hop-recursion
```

BGP4+ NLRIs and next hop attributes

BGP4+ introduces new attributes to handle multiprotocol extensions for BGP.

Multiprotocol BGP (MBGP) is an extension to BGP that enables BGP to carry routing information for multiple address families.

BGP4+ introduces new attributes to handle multiprotocol extensions for BGP:

- Multiprotocol reachable Network Layer Reachability Information (MP_REACH_NLRI): Used to carry the set of reachable destinations, together with the next hop information, to be used for forwarding to these destinations.
- Multiprotocol unreachable NLRI (MP_UNREACH_NLRI): Used to carry the set of unreachable destinations.

MP_REACH_NLRI and MP_UNREACH_NLRI are optional and non-transitive, so that a BGP4+ speaker that does not support the multiprotocol capabilities ignores the information carried in these attributes, and does not pass it to other BGP4+ speakers. A BGP speaker that uses multiprotocol extensions for IPv6 uses the capability advertisement procedures to determine whether the speaker can use multiprotocol extensions with a particular peer.

The next hop information carried in the MP_REACH_NLRI path attribute defines the network layer address of the border router that will be used as the next hop to the destinations listed in the MP_NLRI attribute in the UPDATE message.

MP_REACH_NLRI and MP_UNREACH_NLRI carry IPv6 prefixes.

BGP4+ route reflection

A BGP device can act as a route-reflector client or as a route reflector. You can configure a BGP peer as a route-reflector client from the device that is going to reflect the routes and act as the route reflector using the **neighbor route-reflector-client** command.

When there is more than one route reflector, they should all belong to the same cluster. By default, the value for **cluster-id** is used as the device ID. The device ID can be changed using the **cluster-id** command.

The route-reflector server reflects the routes as follows:

- Routes from the client are reflected to the client as well as to nonclient peers.
- Routes from nonclient peers are reflected only to client peers.

If route-reflector clients are connected in a full IBGP mesh, you can disable client-to-client reflection on the route reflector using the **no client-to-client-reflection** command.

A BGP device advertises only those routes that are preferred ones and are installed into the Routing Table Manager (RTM). When a route cannot be installed into the RTM because the routing table is full, the route reflector may not reflect that route. In cases where the route reflector is not placed directly in the forwarding path, you can configure the route reflector to reflect routes even though those routes are not in the RTM using the **always-propagate** command.

Configuring a cluster ID for a route reflector

The cluster ID can be changed if there is more than one route reflector, so that all route reflectors belong to the same cluster.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   device(config)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

```
device(config-bgp)# local-as 1000
```

5. Enter the **cluster-id** command and specify a value to change the cluster ID of a device from the default device ID.

```
device(config-bgp-router)# cluster-id 321
```

```
device(config-bgp)# cluster-id 321
```

The following example changes the cluster ID of a device from the default device ID to 321.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# cluster-id 321
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# cluster-id 321
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# cluster-id 321
```

Configuring a route reflector client

A BGP peer can be configured as a route reflector client.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

```
device(config-bgp)# local-as 1000
```

5. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

```
device(config-bgp)# address-family ipv6 unicast
```

6. Enter the **neighbor route-reflector-client** command, specifying an IPv6 address, to configure a specified neighbor to be a route reflector client.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 route-reflector-client
```

The following example configures a neighbor with the IPv6 address 2001:db8:e0ff:783a::4 to be a route reflector client.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 route-reflector-client

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 route-reflector-client

device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 route-reflector-client
```

BGP4+ route aggregation

A device can be configured to aggregate routes in a range of networks into a single IPv6 prefix.

By default, a device advertises individual BGP4+ routes for all the networks. The aggregation feature allows you to configure a device to aggregate routes in a range of networks into a single IPv6 prefix. For example, without aggregation, a device will individually advertise routes for networks 2001:db8:0001:0000::/64, 2001:db8:0002:0000::/64, 2001:db8:0003:0000::/64, and so on. You can configure the device to send a single, aggregate route for the networks instead so that the aggregate route would be advertised as 2001:db8::/32 to BGP4 neighbors.

Aggregating routes advertised to BGP neighbors

A device can be configured to aggregate routes in a range of networks into a single IPv6 prefix.

The route-map should already be defined.

You can aggregate BGP4+ routes, for example 2001:db8:0001:0000::/64, 2001:db8:0002:0000::/64, 2001:db8:0003:0000::/64 into a single network prefix: 2001:db8::/24.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

   ```
   device(config)# router bgp
   ```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

   ```
   device(config-bgp)# address-family ipv6 unicast
   ```

5. Enter the **aggregate-address** command to aggregate the routes from a range of networks into a single network prefix.

```
device(config-bgp-ipv6u)# aggregate-address 2001:db8::/32
```

The following example enables a BGP4+ device to advertise the default route and send the default route to a specified neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# aggregate-address 2001:db8::/32

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# aggregate-address 2001:db8::/32

device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# aggregate-address 2001:db8::/32
```

BGP4+ multipath

The BGP4+ multipath feature can be used to enable load-balancing across different paths.

BGP4+ selects only one best path for each IPv6 prefix it receives before installing it in the IP routing table. If you need load-balancing across different paths, you must enable BGP4+ multipath using the **maximum-paths** command under IPv6 address family configuration mode.

IBGP paths and EBGP paths can be exclusively selected, or a combination of IBGP and EBGP paths can be selected.

The following attributes of parallel paths must match for them to be considered for multipathing:

- Weight
- Local Preference
- Origin
- AS-Path Length
- MED
- Neighbor AS (EBGP multipath)
- AS-PATH match (for IBGP multipath)
- IGP metric to BGP next hop

Enabling load-balancing across different paths

The BGP4+ multipath feature can be configured, enabling load-balancing across different paths.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
device(config)# router bgp
```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp)# address-family ipv6 unicast
```

5. Do one of the following:
   - Enter the **maximum-paths** command and specify a value to set the maximum number of BGP4+ shared paths.
   - Enter the **maximum-paths** command using the **use-load-sharing** keyword to set the maximum number of BGP4+ shared paths to that of the value already configured by means of the **ip load-sharing** command.

```
device(config-bgp-ipv6u)# maximum-paths 8
```

or

```
device(config-bgp-ipv6u)# maximum-paths use-load-sharing
```

NOTE
The **ip load-sharing** command has been deprecated. Note the following use cases with respect to legacy configurations and the current release:

1. If the **ip load-sharing** command has been used in a previous release and maximum paths **has not** been configured, then load balancing is influenced only for static routes and load balancing for the routing protocol is not affected.
2. If the **ip load-sharing** command has been used in a previous release and maximum paths **has** been configured, then the lowest configured number of paths takes precedence.
3. If only the **ip load-sharing** command has been used with an existing configuration and maximum paths **has not** been configured, then there is no effect on load sharing for the routing protocol.
4. If only the **ip load-sharing** command has been used with an existing configuration and maximum paths **has** been configured, then the lowest configured number of paths takes precedence. The number of maximum paths can still be adjusted, by means of the **maximum-paths** command, prior to an upgrade to the current release.

The following example sets the maximum number of BGP4+ shared paths to 8.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# maximum-paths 8
```

The following example sets the maximum number of BGP4+ shared paths to that of the value already configured using the **ip load-sharing** command.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# maximum-paths use-load-sharing
```

The following example sets the maximum number of BGP4+ shared paths to 8.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# maximum-paths 8
```

The following example sets the maximum number of BGP4+ shared paths to 8.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# maximum-paths 8
```

The following example sets the maximum number of BGP4+ shared paths to that of the value already configured using the **ip load-sharing** command.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# maximum-paths use-load-sharing
```

Route maps

Route maps must be applied to IPv6 unicast address prefixes in IPv6 address family configuration mode.

By default, route maps that are applied under IPv4 address family configuration mode using the **neighbor route-map** command are applied to only IPv4 unicast address prefixes. To apply route maps to IPv6 unicast address prefixes, the **neighbor route-map** command must be used in IPv6 address family configuration mode. The route maps are applied as the inbound or outbound routing policy for neighbors under the specified address family. Configuring separate route maps under each address family type simplifies managing complicated or different policies for each address family.

Configuring a route map for BGP4+ prefixes

Route maps can be applied to IPv6 unicast address prefixes either as the inbound or outbound routing policy for neighbors under the specified address family.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 prefix-list** command and enter a name to configure an IPv6 prefix list.

   ```
   device(config)# ipv6 prefix-list myprefixlist seq 10 permit 2001:db8::/32
   ```

   The prefix list name, sequence number, and permits packets are specified.

3. Enter the **route-map** command with the **permit** keyword, and specify a route map name, to define the route map and enter route map configuration mode.

   ```
   device(config)# route-map myroutemap permit 10
   ```

4. Enter the **match ipv6 address** command and specify the name of a prefix list.

   ```
   device(config-route-map-myroutemap)# match ipv6 address prefix-list myprefixlist
   device(config-route-map-myroutemap/permit/10)# match ipv6 address prefix-list myprefixlist
   ```

5. Enter the **exit** command to return to global configuration mode.

```
device(config-route-map-myroutemap)# exit
device(config-route-map-myroutemap/permit/10)# exit
```

6. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

7. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
device(config-bgp)# local-as 1000
```

8. Enter the **neighbor remote-as** command, specifying an IPv6 address, to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
```

9. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp)# address-family ipv6 unicast
```

10. Enter the **neighbor activate** command to enable the exchange of information with the neighbor.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
```

11. Enter the **neighbor route-map** command and specify the **out** keyword to apply a route map to outgoing routes.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out myroutemap
```

The following example applies a route map, “myroutemap”, as the outbound routing policy for a neighbor.

```
device# configure terminal
device(config)# ipv6 prefix-list myprefixlist seq 10 permit 2001:db8::/32
device(config)# route-map myroutemap permit 10
device(config-route-map-myroutemap)# match ipv6 address prefix-list myprefixlist
device(config-route-map-myroutemap)# exit
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out myroutemap

device# configure terminal
device(config)# ipv6 prefix-list myprefixlist seq 10 permit 2001:db8::/32
device(config)# route-map myroutemap permit 10
device(config-route-map-myroutemap/permit/10)# match ipv6 address prefix-list myprefixlist
device(config-route-map-myroutemap/permit/10)# exit
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out myroutemap
```

Redistributing prefixes into BGP4+

Various routes can be redistributed into BGP.

Static, connected, and OSPF routes can be redistributed into BGP. This task redistributes OSPFv3 routes into BGP4+.

Static, connected, OSPF, ISIS, and RIPng routes can be redistributed into BGP. This task redistributes RIPng routes into BGP4+.

Static, connected, OSPF, and ISIS routes can be redistributed into BGP. This task redistributes OSPFv3 routes into BGP4+.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

   ```
   device(config)# router bgp
   ```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

   ```
   device(config-bgp)# address-family ipv6 unicast
   ```

5. Enter the **redistribute** command using the **ospf** and **external1** keywords to redistribute IPv6 OSPF external type 1 routes.

   ```
   device(config-bgp-ipv6u)# redistribute ospf match external1
   ```

6. Enter the **redistribute** command using the **rip** keyword to redistribute IPv6 RIP routes.

   ```
   device(config-bgp-ipv6u)# redistribute rip
   ```

The following example redistributes OSPF external type 1 routes into BGP4+.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# redistribute ospf match external1
```

The following example redistributes RIPng prefixes into BGP4+.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# redistribute rip
```

The following example redistributes OSPF external type 1 routes into BGP4+.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# redistribute ospf match external1
```

Redistributing routes into BGP4+

Various routes can be redistributed into BGP. This task redistributes connected routes into BGP.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

   ```
   device(config)# router bgp
   ```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

   ```
   device(config-bgp)# address-family ipv6 unicast
   ```

5. Enter the **redistribute** command using the **connected** keyword to redistribute connected routes into BGP4+.

   ```
   device(config-bgp-ipv6u)# redistribute connected
   ```

The following example redistributes connected routes into BGP4+.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# redistribute connected

device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# redistribute connected

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# redistribute connected
```

Specifying the weight added to BGP4+ received routes

The weight that the device adds to received routes can be specified. The following task changes the weight from the default for routes that are received from a specified BGP4+ neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
device(config)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
device(config-bgp)# local-as 65520
```

5. Enter the **address-family ipv6 unicast** command to enter address family IPv6 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp)# address-family ipv6 unicast
```

6. Enter the **neighbor weight** command and specify an *ipv6 address* and a weight value to specify a weight that the device adds to routes that are received from the specified BGP4+ neighbor.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 weight 200
```

The following example specifies a weight of 200 that the device adds to routes that are received from the specified BGP4+ neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 weight 200

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 weight 200

device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 weight 200
```

Enabling BGP4+ in a non-default VRF

When BGP4+ is enabled in a non-default VRF instance, the device enters BGP address-family IPv6 unicast VRF configuration mode. Several commands can then be accessed that allow the configuration of BGP4+ for the non-default VRF instance.

A non-default VRF instance has been configured.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing globally.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **address-family ipv6 unicast** command with the **vrf** parameter, and specify a VRF name, to enter BGP address-family IPv6 unicast VRF configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast vrf green
```

```
device(config-bgp)# address-family ipv6 unicast vrf green
```

The following example enables BGP address-family IPv6 unicast VRF configuration mode where several commands can be accessed that allow the configuration of BGP4+ for the non-default VRF instance..

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast vrf green
device(config-bgp-ipv6u-vrf)#

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast vrf green
device(config-bgp-ipv6u-vrf)#

device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv6 unicast vrf green
device(config-bgp-ipv6u-vrf)#
```

BGP4+ outbound route filtering

The BGP4+ Outbound Route Filtering Capability (ORF) feature is used to minimize the number of BGP updates sent between BGP peers.

When the ORF feature is enabled, unwanted routing updates are filtered out, reducing the amount of system resources required for generating and processing routing updates. The ORF feature is enabled through the advertisement of ORF capabilities to peer routers. The locally configured BGP4+ inbound prefix filters are sent to the remote peer so that the remote peer applies the filter as an outbound filter for the neighbor.

The ORF feature can be configured with send and receive ORF capabilities. The local peer advertises the ORF capability in send mode, indicating that it will accept a prefix list from a neighbor and apply the prefix list to locally configured ORFs. The local peer exchanges the ORF capability in send mode with a remote peer for a prefix list that is configured as an inbound filter for that peer locally. The remote peer only sends the first update once it receives a ROUTEREFRESH request or BGP ORF with IMMEDIATE from the peer. The local and remote peers exchange updates to maintain the ORF on each router.

Configuring BGP4+ outbound route filtering

The BGP4+ Outbound Route Filtering (ORF) prefix list capability can be configured in receive mode, send mode, or both send and receive modes, minimizing the number of BGP updates exchanged between BGP peers.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
device(config)# router bgp
```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp)# address-family ipv6 unicast
```

5. Enter the **neighbor activate** command, specifying an IPv6 address, to add a neighbor.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
```

6. Enter the **neighbor prefix-list** command, specify an IPv6 address and the **in** keyword to filter the incoming route updates from a specified BGP neighbor.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 prefix-list myprefixlist in
```

7. Do one of the following:
   - Enter the **neighbor capability orf prefixlist** command and specify the **send** keyword to advertise ORF send capabilities.

     ```
     device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist send
     ```

   - Enter the **neighbor capability orf prefixlist** command and specify the **receive** keyword to advertise ORF receive capabilities.

     ```
     device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist receive
     ```

   - Enter the **neighbor capability orf prefixlist** command to configure ORF capability in both send and receive modes.

     ```
     device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist
     ```

The following example configures ORF in receive mode.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist receive

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist receive

device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist receive
```

The following example configures ORF in send mode.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 prefix-list myprefixlist in
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist send

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 prefix-list myprefixlist in
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist send

device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 prefix-list myprefixlist in
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist send
```

The following example configures ORF in both send and receive modes.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 prefix-list myprefixlist in
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 prefix-list myprefixlist in
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist

device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 prefix-list myprefixlist in
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist
```

BGP4+ confederations

A large autonomous system (AS) can be divided into multiple subautonomous systems and grouped into a single BGP4+ confederation.

Each subautonomous system must be uniquely identified within the confederation AS by a subautonomous system number. Within each subautonomous system, all the rules of internal BGP (IBGP) apply. For example, all BGP routers inside the subautonomous system must be fully meshed. Although EBGP is used between subautonomous systems, the subautonomous systems within the confederation exchange routing information like IBGP peers. Next hop, Multi Exit Discriminator (MED), and local preference information is preserved when crossing subautonomous system boundaries. To the outside world, a confederation looks like a single AS.

The AS path list is a loop-avoidance mechanism used to detect routing updates leaving one subautonomous system and attempting to re-enter the same subautonomous system. A routing update attempting to re-enter a subautonomous system it originated from is detected because the subautonomous system sees its own subautonomous system number listed in the update's AS path.

Configuring BGP4+ confederations

BGP4+ confederations, composed of multiple subautonomous systems, can be created.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   device(config)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   ```

5. Enter the **confederation identifier** command and specify an ASN to configure a BGP confederation identifier.

   ```
   device(config-bgp-router)# confederation identifier 100
   ```

6. Enter the **confederation peers** command and specify as many ASNs as needed to list all BGP peers that will belong to the confederation.

   ```
   device(config-bgp-router)# confederation peers 65520 65521 65522
   ```

The following example creates a confederation with the confederation ID “100” and adds three subautonomous systems to the confederation.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# confederation identifier 100
device(config-bgp-router)# confederation peers 65520 65521 65522

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# confederation identifier 100
device(config-bgp-router)# confederation peers 65520 65521 65522
```

BGP4+ extended community

The BGP4+ extended community feature filters routes based on a regular expression specified when a route has multiple community values in it.

A BGP community is a group of destinations that share a common property. Community information identifying community members is included as a path attribute in BGP UPDATE messages. You can perform actions on a group using community and extended community attributes to trigger routing decisions. All communities of a particular type can be filtered out, or certain values can be specified for a particular type of community. You can also specify whether a particular community is transitive or non-transitive across an autonomous system (AS) boundary.

An extended community is an 8-octet value and provides a larger range for grouping or categorizing communities. BGP extended community attributes are specified in RFC 4360.

You define the extended community list using the **ip extcommunity-list** command. The extended community can then be matched or applied to the neighbor through the route map. The route map must be applied on the neighbor to which routes need to carry the extended community attributes. The "send-community" should be enabled for the neighbor configuration to start including the attributes while sending updates to the neighbor.

Defining BGP4+ extended communities

In order to apply a BGP4+ extended community filter, a BGP4+ extended community filter must be defined.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ip extcommunity-list** command and specify a number to set a BGP extended community filter.

   ```
   device(config-rbridge-id-122)# ip extcommunity-list 1 permit soo 123:2
   ```

4. Enter the **ip extcommunity-list** command with the **standard** keyword and specify a name to set a standard BGP extended community filter.

   ```
   device(config)# ip extcommunity-list standard mylist permit soo 123:2
   ```

5. Enter the **route-map** *name* command to create and define a route map and enter route-map configuration mode.

   ```
   device(config-rbridge-id-122)# route-map extComRmap permit 10
   device(config)# route-map extComRmap permit 10
   ```

   Permits a matching pattern.

6. Enter the **match extcommunity** command and specify an extended community list.

   ```
   device(config-route-map-extComRmap/permit/10)# match extcommunity 1
   device(config-route-map-extComRmap/permit/10)# match extcommunity mylist
   ```

7. Enter the **exit** command.

   ```
   device(config-route-map-extComRmap/permit/10)# exit
   ```

8. Enter the **route-map***name* command to define a route map and enter route-map configuration mode.

   ```
   device(config-rbridge-id-122)# route-map sendExtComRmap permit 10
   device(config)# route-map sendExtComRmap permit 10
   ```

   Permits a matching pattern.

9. Enter the **set extcommunity** command and specify the **rt** *extcommunity value* keyword to specify the route target (RT) extended community attribute.

   ```
   device(config-route-map-sendExtComRmap/permit/10)# set extcommunity rt 3:3
   ```

10. Enter the **set extcommunity** command and specify the **soo** *extcommunity value* keyword to the site of origin (SOO) extended community attribute.

```
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity soo 2:2
```

The following example configures an extended community ACL called “extended”, defines a route map, and permits and sets a matching pattern.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip extcommunity-list 1 permit soo 123:2
device(config-rbridge-id-122)# route-map extComRmap permit 10
device(config-route-map-extComRmap/permit/10)# match extcommunity 1
device(config-route-map-extComRmap/permit/10)# exit
device(config-rbridge-id-122)# route-map sendExtComRmap permit 10
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity rt 3:3
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity soo 2:2

device# configure terminal
device(config)# ip extcommunity-list standard mylist permit soo 123:2
device(config)# route-map extComRmap permit 10
device(config-route-map-extComRmap/permit/10)# match extcommunity mylist
device(config-route-map-extComRmap/permit/10)# exit
device(config)# route-map sendExtComRmap permit 10
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity rt 3:3
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity soo 2:2
```

Applying a BGP4+ extended community filter

A BGP4+ extended community filter can be applied .

BGP4+ communities must already be defined.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip extcommunity-list** command and specify a number to set a BGP extended community filter.

```
device(config-rbridge-id-122)# ip extcommunity-list 1 permit rt 123:2
```

```
device(config)# ip extcommunity-list 1 permit rt 123:2
```

4. Enter the **ip extcommunity-list** command with the **standard** keyword and specify a name to set a standard BGP extended community filter.

```
device(config)# ip extcommunity-list standard mylist permit rt 123:2
```

5. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

6. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

```
device(config-bgp)# local-as 1000
```

7. Enter the **neighbor remote-as** command, specifying an IPv6 address, to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
```

```
device(config-bgp)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
```

8. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

```
device(config-bgp)# address-family ipv6 unicast
```

9. Enter the **neighbor activate** command to enable the exchange of information with the neighbor.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
```

10. Enter the **neighbor route-map** command and specify the **in** keyword to apply a route map to incoming routes.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map in extComRmapt
```

11. Enter the **neighbor route-map** command and specify the **out** keyword to apply a route map to outgoing routes.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out sendExtComRmap
```

12. Enter the **neighbor send-community** command and specify the **both** keyword to enable the sending of standard and extended attributes in updates to the specified BGP neighbor.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 send-community both
```

The following example applies a BGP4+ extended community filter.

```
device# configure terminal
device(config)# ip extcommunity-list 1 permit rt 123:2
device(config)# router bgp
device(config-bgp)# local-as 1000
device(config-bgp)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map in extComRmap
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out sendExtComRmap
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 send-community both

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip extcommunity-list 1 permit rt 123:2
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map in extComRmap
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out sendExtComRmap
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 send-community both

device# configure terminal
device(config)# ip extcommunity-list standard mylist permit rt 123:2
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map in extComRmap
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out sendExtComRmap
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 send-community both
```

BGP4+ graceful restart

BGP4+ graceful restart (GR) allows for restarts where BGP neighboring devices participate in the restart, helping to ensure that no route and topology changes occur in the network for the duration of the restart.

The GR feature provides a routing device with the capability to inform its neighbors when it is performing a restart.

When a BGP session is established, GR capability for BGP is negotiated by neighbors through the BGP OPEN message. If the neighbor also advertises support for GR, GR is activated for that neighbor session. If both peers do not exchange the GR capability, the session is not GR-capable. If the BGP session is lost, the BGP peer router, known as a GR helper, marks all routes associated with the device as "stale" but continues to forward packets to these routes for a set period of time. The restarting device also continues to forward packets for the duration of the graceful restart. When the graceful restart is complete, routes are obtained from the helper so that the device is able to quickly resume full operation.

When the GR feature is configured on a device, both helper router and restarting router functionalities are supported. It is not possible to disable helper functionality explicitly.

GR is disabled by default and can be enabled in both IPv4 and IPv6 address families. When the GR timer expires, the BGP RASlog message is triggered.

> **NOTE**
> GR is not supported in BGP IPv6 address-family multicast configuration mode.

> **NOTE**
> BGP4+ GR can be configured for a global routing instance or for a specified VRF instance.

> **NOTE**
> BGP4+ GR is fully supported by MLX Series and XMR Series devices. The CER 2000 Series and CES 2000 Series devices only support GR helper mode.

Configuring BGP4+ graceful restart

The graceful restart (GR) feature can be configured on a routing device, providing it with the capability to inform its neighbors when it is performing a restart.

> **NOTE**
> High availability (HA) requires GR to be enabled.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   device(config)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
device(config-bgp)# local-as 1000
```

5. Enter the **neighbor remote-as** command, specifying an IPv6 address, to specify the autonomous system ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp)# neighbor 1000::1 remote-as 2
```

6. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp)# address-family ipv6 unicast
```

7. Enter the **neighbor activate** command to add a neighbor.

```
device(config-bgp-ipv6u)# neighbor 1000::1 activate
```

8. Enter the **graceful-restart** command to enable the graceful restart feature.

```
device(config-bgp-ipv6u)# graceful-restart
```

9. Do any of the following:
   - Enter the **graceful-restart** command using the **purge-time** keyword to overwrite the default purge-time value.

     ```
     device(config-bgp-ipv6u)# graceful-restart purge-time 300
     ```

   - Enter the **graceful-restart** command using the **restart-time** keyword to overwrite the default restart-time advertised to graceful restart-capable neighbors.

     ```
     device(config-bgp-ipv6u)# graceful-restart restart-time 180
     ```

   - Enter the **graceful-restart** command using the **stale-routes-time** keyword to overwrite the default amount of time that a helper device will wait for an EOR message from a peer.

     ```
     device(config-bgp-ipv6u)# graceful-restart stale-routes-time 100
     ```

The following example enables the graceful restart feature.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart

device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1
device(config-bgp)# neighbor 1000::1 remote-as 2
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart
```

The following example enables the graceful restart feature and sets the purge time to 300 seconds, overwriting the default value.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart purge-time 300

device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1
device(config-bgp)# neighbor 1000::1 remote-as 2
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart purge-time 300

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart purge-time 300
```

The following example enables the graceful restart feature and sets the restart time to 180 seconds, overwriting the default value.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart restart-time 180
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1
device(config-bgp)# neighbor 1000::1 remote-as 2
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart restart-time 180
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart restart-time 180
```

The following example enables the graceful restart feature and sets the stale-routes time to 100 seconds, overwriting the default value.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart stale-routes-time 100
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 1
device(config-bgp)# neighbor 1000::1 remote-as 2
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart stale-routes-time 100
```

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart stale-routes-time 100
```

Use the **clear ipv6 bgp neighbor** command with the **all** parameter for the changes to the graceful restart parameters to take effect immediately.

Auto shutdown of BGP neighbors on initial configuration

The auto shutdown of BGP neighbors on initial configuration feature prevents a BGP peer from attempting to establish connections with remote peers when the BGP peer is first configured. Sessions are only initiated when the entire configuration for the BGP peer is complete.

When the auto shutdown of BGP neighbors on initial configuration feature is enabled, the value of the shutdown parameter for any existing configured neighbor is not changed. Any new BGP neighbor configured after the feature is enabled has the shutdown state set to the configured value. When the feature is enabled and a new BGP neighbor is configured, the shutdown parameter of the BGP neighbor is automatically set to enabled. When all the BGP neighbor parameters are configured and it is ready to start the establishment of BGP session with the remote peer, the shutdown parameter of the BGP neighbor is then disabled.

Any new neighbor configured and added to a peer group has the shutdown flag enabled by default. Additionally, any new neighbor configured with the autonomous system (AS) in which that remote neighbor resides specified using the **neighbor remote-as** command has the shutdown flag enabled by default.

Configuring auto shutdown of BGP neighbors on initial configuration

Follow this procedure to enable auto shutdown of BGP neighbors on initial configuration.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp)# local-as 65520
   device(config-bgp-router)# local-as 65520
   ```

5. Enter the **auto-shutdown-new-neighbors** command to prevent the BGP peer from attempting to establish connections with remote peers until all BGP neighbor parameters are configured.

   ```
   device(config-bgp)# auto-shutdown-new-neighbors
   device(config-bgp-router)# auto-shutdown-new-neighbors
   ```

The following example enables auto shutdown of BGP neighbors on initial configuration.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# auto-shutdown-new-neighbors

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# auto-shutdown-new-neighbors

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# auto-shutdown-new-neighbors
```

Disabling the BGP4+ peer shutdown state

When the auto shutdown of BGP4+ neighbors on initial configuration feature is enabled, a BGP4+ peer is prevented from attempting to establish connections with remote peers when the BGP4+ peer is first configured. Once all of the configuration parameters for the peer are complete, you can start the BGP4+ session establishment process and disable the peer shutdown state. Follow this procedure to disable the peer shutdown state.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp)# local-as 65520
   device(config-bgp-router)# local-as 65520
   ```

5. Enter the **no neighbor shutdown** command, specifying an IPv6 address, to disable the peer shutdown state and begin the BGP4+ session establishment process.

   ```
   device(config-bgp)# no neighbor 2001:2018:8192::125 shutdown
   device(config-bgp-router)# no neighbor 2001:2018:8192::125 shutdown
   ```

The following example disables the peer shutdown state and begins the BGP4+ session establishment process.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# no neighbor 2001:2018:8192::125 shutdown

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# no neighbor 2001:2018:8192::125 shutdown

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# no neighbor 2001:2018:8192::125 shutdown
```

Generalized TTL Security Mechanism support

Generalized TTL Security Mechanism (GTSM) is a lightweight security mechanism that protects external Border Gateway Protocol (eBGP) peering sessions from CPU utilization-based attacks using forged IP packets. GTSM prevents attempts to hijack the eBGP peering session by a host on a network segment that is not part of either BGP network, or by a host on a network segment that is not between the eBGP peers.

GTSM is enabled by configuring a minimum Time To Live (TTL) value for incoming IP packets received from a specific eBGP peer. BGP establishes and maintains the session only if the TTL value in the IP packet header is equal to or greater than the TTL value configured for the peering session. If the value is less than the configured value, the packet is silently discarded and no Internet Control Message Protocol (ICMP) message is generated.

In the case of directly connected neighbors, the device expects the BGP control packets received from the neighbor to have a TTL value of either 254 or 255. For multihop peers, the device expects the TTL for BGP control packets received from the neighbor to be greater than or equal to 255, minus the configured number of hops to the neighbor. If the BGP control packets received from the neighbor do not have the expected value, the device drops them.

When GTSM protection is enabled, BGP control packets sent by the device to a neighbor have a Time To Live (TTL) value of 255. In addition, the device expects the BGP control packets received from the neighbor to have a TTL value of either 254 or 255. For multihop peers, the device expects the TTL for BGP control packets received from the neighbor to be greater than or equal to 255, minus the configured number of hops to the neighbor. If the BGP control packets received from the neighbor do not have the anticipated value, the device drops them.

For more information on GTSM protection, refer to RFC 3682.

Assumptions and limitations

- GTSM is supported for both directly connected peering sessions and multihop eBGP peering sessions.
- GTSM is supported for eBGP only.
- GTSM does not protect the integrity of data sent between eBGP peers and does not validate eBGP peers through any authentication method.
- GTSM validates only the locally configured TTL count against the TTL field in the IP packet header.
- GTSM should be configured on each participating device to maximize the effectiveness of this feature.
- When GTSM is enabled, the eBGP session is secured in the incoming direction only and has no effect on outgoing IP packets or the remote device.

Configuring GTSM for BGP4+

Generalized TTL Security Mechanism (GTSM) can be configured to protect external Border Gateway Protocol (eBGP) peering sessions .

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   device(config-bgp)# local-as 65520
   ```

5. Enter the **address-family ipv6 unicast** command to enter IPv6 address family unicast configuration mode.

   ```
   device(config-bgp)# address-family ipv6 unicast
   ```

6. Enter the **neighbor remote-as** command, specifying an IPv6 address, to add a neighbor.

   ```
   device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 remote-as 2
   device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
   ```

7. Enter the **neighbor ebgp-btsh** command, specifying an IPv6 address, to enable GTSM.

   ```
   device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 ebgp-btsh
   device(config-bgp-router)# neighbor 2001:2018:8192::125 ebgp-btsh
   ```

The following example enables GTSM between a device and a neighbor with the IPv6 address 2001:2018:8192::125.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# local-as 65520
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 remote-as 2
device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 ebgp-btsh

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
device(config-bgp-router)# neighbor 2001:2018:8192::125 ebgp-btsh

device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
device(config-bgp-router)# neighbor 2001:2018:8192::125 ebgp-btsh
```

Disabling the BGP AS_PATH check function

A device can be configured so that the AS_PATH check function for routes learned from a specific location is disabled, and routes that contain the recipient BGP speaker's AS number are not rejected.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   device(config)# router bgp
   ```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   device(config-bgp)# address-family ipv6 unicast
   ```

5. Enter the **neighbor allowas-in** command with an IPv6 address and specify a **number** to disable the BGP AS_PATH check function, and specify the number of times that the AS path of a received route may contain the recipient BGP speaker's AS number and still be accepted.

   ```
   device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 allowas-in 3
   ```

The following example specifies that the AS path of a received route may contain the recipient BGP speaker's AS number three times and still be accepted.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 allowas-in 3

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 allowas-in 3

device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 allowas-in 3
```

Displaying BGP4+ statistics

Various **show ipv6 bgp** commands verify information about BGP4+ configurations.

Use one or more of the following commands to verify BGP4+ information. The commands do not have to be entered in this order.

1. Enter the **show ipv6 bgp summary** command.

```
device# show ipv6 bgp summary

  BGP4 Summary
  Router ID: 107.1.1.8   Local AS Number: 100
  Confederation Identifier: not configured
  Confederation Peers:
  Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
  Number of Neighbors Configured: 1, UP: 0
  Number of Routes Installed: 1, Uses 96 bytes
  Number of Routes Advertising to All Neighbors: 1 (1 entries), Uses 60 bytes
  Number of Attribute Entries Installed: 1, Uses 104 bytes
  Neighbor Address  AS#         State     Time     Rt:Accepted Filtered Sent     ToSend
  1:2::3            100         CONN      0h 0m18s    0        0        0        1
```

   This example output gives summarized BGP4+ information.

2. Enter the **show ipv6 bgp attribute-entries** command.

```
device# show ipv6 bgp attribute-entries

        Total number of BGP Attribute Entries: 1
1       Next Hop  : ::                                          MED      :0
Origin:INCOMP
        Originator:0.0.0.0            Cluster List:None
        Aggregator:AS Number :0            Router-ID:0.0.0.0          Atomic:None
        Local Pref:100               Communities:Internet
        AS Path   : (length 0)
           AsPathLen: 0  AsNum: 0, SegmentNum: 0, Neighboring As: 0, Source As 0
        Address: 0x0b456c4c  Hash:876 (0x03000000)
        Links: 0x00000000, 0x00000000
        Reference Counts: 1:0:1, Magic: 2
```

   This example shows information about an route-attribute entry that is stored in device memory.

3. Enter the **show ipv6 bgp peer-group** command.

```
device# show ipv6 bgp peer-group

1   BGP peer-group is pg
      Address family : IPV4 Unicast
        activate
      Address family : IPV6 Unicast
        no activate
    Members:
       IP Address: 1.1.1.1, AS: 100
       IP Address: 1::1, AS: 100

2   BGP peer-group is pg6
      Address family : IPV4 Unicast
        activate
      Address family : IPV6 Unicast
        no activate
    Currently there are no members.
```

   This example shows output for two peer groups, called “pg” and “pg6”.

4. Enter the **show ipv6 bgp routes** command.

```
device# show ipv6 bgp routes

Total number of BGP Routes: 1
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      107:1:1::/64       ::              0          100        32768  BL
         AS_PATH:
```

This example shows general BGP4+ route information.

5. Enter the **show ipv6 bgp routes** command, using the **summary** keyword.

```
device# show ipv6 bgp routes summary

  Total number of BGP routes (NLRIs) Installed     : 1
  Distinct BGP destination networks                : 1
  Filtered bgp routes for soft reconfig            : 0
  Routes originated by this router                 : 1
  Routes selected as BEST routes                   : 1
  Routes Installed as BEST routes                  : 1
  BEST routes not installed in IP forwarding table : 0
  Unreachable routes (no IGP route for NEXTHOP)    : 0
  IBGP routes selected as best routes              : 0
  EBGP routes selected as best routes              : 0
  BEST routes not valid for IP forwarding table    : 0
```

This example shows summarized BGP4+ route information.

Displaying BGP4+ statistics

Various **show ipv6 bgp** commands verify information about BGP4+ configurations.

Use one or more of the following commands to verify BGP4+ information. The commands do not have to be entered in this order.

1. Enter the **show ipv6 bgp summary** command.

```
device# show ipv6 bgp summary

  BGP4 Summary
  Router ID: 30.1.1.8   Local AS Number: 4001183001
  Confederation Identifier: not configured
  Confederation Peers:
  Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
  Number of Neighbors Configured: 3, UP: 3
  Number of Routes Installed: 200, Uses 19200 bytes
  Number of Routes Advertising to All Neighbors: 400 (200 entries), Uses 12000 bytes
  Number of Attribute Entries Installed: 1, Uses 104 bytes
  Neighbor Address  AS#         State     Time     Rt:Accepted Filtered Sent     ToSend
  fd80:3001:4031:1::225
                    4031        ESTAB     8h12m 5s    200      0        0        0
  fdcd:3001:3009:1::113
                    4001133001  ESTAB     8h 6m47s    0        0        200      0
  fdcd:3001:3009:1::114
                    4001143001  ESTAB     6h40m33s    0        0        200      0
```

This example output gives summarized BGP4+ information.

2. Enter the **show ipv6 bgp attribute-entries** command.

```
device# show ipv6 bgp attribute-entries

        Total number of BGP Attribute Entries: 1
1       Next Hop  : fd80:3001:4031:1::225                     MED      :0
Origin:IGP
        Originator:0.0.0.0           Cluster List:None
        Aggregator:AS Number :0            Router-ID:0.0.0.0          Atomic:None
        Local Pref:100               Communities:Internet
        AS Path   :4031 (length 4)
           AsPathLen: 1  AsNum: 1, SegmentNum: 1, Neighboring As: 4031, Source As 4031
        Address: 0x0a5e1e18  Hash:6995 (0x01002080)
        Links: 0x00000000, 0x00000000
        Reference Counts: 200:0:400, Magic: 13
```

This example shows information about an route-attribute entry that is stored in device memory.

3. Enter the **show ipv6 bgp peer-group** command.

```
device# show ipv6 bgp peer-group

1   BGP peer-group is pg_vrf11, Remote AS: 4001143011
    Description: bgp_vrf4@-4001183011
       MD5 Password: $MiJTfXJVRzMxTTNRUVpaVzhRWlo=
      Address family : IPV4 Unicast
        activate
      Address family : IPV6 Unicast
        no activate
    Currently there are no members.

2   BGP peer-group is pg_vrf12, Remote AS: 4001143021
    Description: bgp_vrf4@-4001183021
       MD5 Password: $MiJTfXJVRzMxTTNRUVpaVzhRfFo=
      Address family : IPV4 Unicast
        activate
      Address family : IPV6 Unicast
        no activate
    Currently there are no members.
```

This example shows output for two peer groups, called “pg” and “pg6”.

4. Enter the **show ipv6 bgp routes** command.

```
device# show ipv6 bgp routes

Total number of BGP Routes: 6
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      fd70::/128         fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 4031
2      fd70::1/128        fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 4031
3      fd70::2/128        fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 4031
4      fd70::3/128        fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 4031
5      fd70::4/128        fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 4031
6      fd70::5/128        fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 4031
```

This example shows general BGP4+ route information.

5. Enter the **show ipv6 bgp routes** command, using the **summary** keyword.

```
device# show ipv6 bgp routes summary

  Total number of BGP routes (NLRIs) Installed       : 200
  Distinct BGP destination networks                  : 200
  Filtered bgp routes for soft reconfig              : 0
  Routes originated by this router                   : 0
  Routes selected as BEST routes                     : 200
  Routes Installed as BEST routes                    : 200
  BEST routes not installed in IP forwarding table   : 0
  Static routes not installed in IP forwarding table : 0
  Unreachable routes (no IGP route for NEXTHOP)      : 0
  IBGP routes selected as best routes                : 0
  EBGP routes selected as best routes                : 200
  BEST routes not valid for IP forwarding table      : 0
```

This example shows summarized BGP4+ route information.

Displaying BGP4+ neighbor statistics

Various **show ipv6 bgp neighbor** commands verify information about BGP4+ neighbor configurations.

Use one or more of the following commands to verify BGP4+ neighbor information. The commands do not have to be entered in this order.

1. Enter the **show ipv6 bgp neighbors** command.

```
device# show ipv6 bgp neighbors

    Total number of BGP Neighbors: 1
1   IP Address: 1:2::3, AS: 100 (IBGP), RouterID: 0.0.0.0, VRF: default-vrf
    State: CONNECT, Time: 0h3m3s, KeepAliveTime: 60, HoldTime: 180
    Minimal Route Advertisement Interval: 0 seconds
    Messages:    Open    Update  KeepAlive Notification Refresh-Req
       Sent    : 0       0       0         0            0
       Received: 0       0       0         0            0
    Last Connection Reset Reason:Unknown
    Notification Sent:     Unspecified
    Notification Received: Unspecified
    Neighbor NLRI Negotiation:
      Peer configured for IPV6 unicast  Routes
    Neighbor ipv6 MPLS Label Capability Negotiation:
    Neighbor AS4 Capability Negotiation:
    Outbound Policy Group:
       ID: 2, Use Count: 3
       Last update time was 172 sec ago
Error: TCP status not available
```

This example output gives summarized information about a BGP4+ neighbor.

2. Enter the **show ipv6 bgp neighbors advertised-routes** command.

```
device# show ipv6 bgp neighbors 123::3 advertised-routes

       There are 5 routes advertised to neighbor 123::3
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      110:110:110::/64   123::2           0                     0      BE
         AS_PATH: 222 111
2      110:110:110:1::/64 123::2           0                     0      BE
         AS_PATH: 222 111
3      110:110:110:2::/64 123::2           0                     0      BE
         AS_PATH: 222 111
4      110:110:110:3::/64 123::2           0                     0      BE
         AS_PATH: 222 111
5      110:110:110:4::/64 123::2           0                     0      BE
         AS_PATH: 222 111
```

This example shows information about all the routes the BGP4+ networking device advertised to the neighbor.

3. Enter the **show ipv6 bgp neighbors last-packet-with-error** command.

```
device# show ipv6 bgp neighbors 123::3 last-packet-with-error

 Received Message Length: 45
 BGP Message:
  0xffffffff  0xffffffff  0xffffffff  0xffffffff  0x002d0104
  0x014b00b4  0x09090909  0x10020601  0x04020000  0x01020202
  0x00020280  0x00

 BGP Header
  Marker:   0xffffffff  0xffffffff  0xffffffff  0xffffffff
  Message Length: (0x002d) 45
  Message Type: (0x01) OPEN

 OPEN Message
 Version: (0x04) 4
 AS Number: (0x014b) 331
 Hold Time: (0x00b4) 180
 BGP Identifier: (0x09090909) 9.9.9.9
 Optional Parameter length: (0x10) 16

 OPEN message optional parameters
  Parameter Type: (0x02) Capability
  Parameter Length: (0x06) 6
   Capability Type: (0x01) MULTIPROTOCOL EXTENSIONS
   Capability Length: (0x04) 4
   AFI: (0x0200) Unknown(512)
   Reserved: (0x00) 0
   SAFI: (0x01) Unicast

  Parameter Type: (0x02) Capability
  Parameter Length: (0x02) 2
   Capability Type: (0x02) ROUTE REFRESH(new)
   Capability Length: (0x00) 0

  Parameter Type: (0x02) Capability
  Parameter Length: (0x02) 2
   Capability Type: (0x80) ROUTE REFRESH(old)
   Capability Length: (0x00) 0
```

This example shows information about the last packet that contained an error from any of a device's neighbors.

4. Enter the **show ipv6 bgp neighbors received-routes** command.

```
device# show ipv6 bgp neighbors 160:160:160::10 received-routes

       There are 5 received routes from neighbor 160:160:160::10
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      110:110:110::/64   160:160:160::10  0         100        0      BE
         AS_PATH: 111
2      110:110:110:1::/64 160:160:160::10  0         100        0      BE
         AS_PATH: 111
3      110:110:110:2::/64 160:160:160::10  0         100        0      BE
         AS_PATH: 111
4      110:110:110:3::/64 160:160:160::10  0         100        0      BE
         AS_PATH: 111
5      110:110:110:4::/64 160:160:160::10  0         100        0      BE
         AS_PATH: 111
```

This example lists all route information received in route updates from BGP4+ neighbors of the device since the soft-reconfiguration feature was enabled.

5. Enter the **show ipv6 bgp neighbors rib-out-routes** command.

```
device# show ipv6 bgp neighbors 123::3 rib-out-routes

       There are 5 RIB_out routes for neighbor 123::3
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      110:110:110::/64   160:160:160::10  0         100        0      BE
         AS_PATH: 111
2      110:110:110:1::/64 160:160:160::10  0         100        0      BE
         AS_PATH: 111
3      110:110:110:2::/64 160:160:160::10  0         100        0      BE
         AS_PATH: 111
4      110:110:110:3::/64 160:160:160::10  0         100        0      BE
         AS_PATH: 111
5      110:110:110:4::/64 160:160:160::10  0         100        0      BE
         AS_PATH: 111
```

This example shows information about BGP4+ outbound RIB routes.

Displaying BGP4+ neighbor statistics

Various **show ipv6 bgp neighbor** commands verify information about BGP4+ neighbor configurations.

Use one or more of the following commands to verify BGP4+ neighbor information. The commands do not have to be entered in this order.

1. Enter the **show ipv6 bgp neighbors** command.

```
device# show ipv6 bgp neighbors

    Total number of BGP Neighbors: 3
1   IP Address: fd80:3001:4031:1::225, AS: 4031 (EBGP), RouterID: 40.31.1.230, VRF: v3001
    State: ESTABLISHED, Time: 8h17m2s, KeepAliveTime: 30, HoldTime: 90
       KeepAliveTimer Expire in 1 seconds, HoldTimer Expire in 60 seconds
    Minimal Route Advertisement Interval: 0 seconds
    Messages:    Open    Update  KeepAlive Notification Refresh-Req
       Sent    : 1       0       1110      0            0
       Received: 1       3       995       0            0
    Last Update Time: NLRI              Withdraw                NLRI                Withdraw
                  Tx: ---               ---                 Rx: 8h17m1s             ---
    Last Connection Reset Reason:User Reset Peer Session
    Notification Sent:     Unspecified
    Notification Received: Unspecified
    Neighbor NLRI Negotiation:
      Peer Negotiated IPV6 unicast  capability
      Peer configured for IPV6 unicast  Routes
    Neighbor AS4 Capability Negotiation:
      Peer configured for AS4  capability
    Outbound Policy Group:
       ID: 5, Use Count: 7
       Byte Sent:   21143, Received: 0
       Local host:  fd80:3001:4031:1::118, Local  Port: 8084
       Remote host: fd80:3001:4031:1::225, Remote Port: 179
…
```

This example output gives summarized information about a BGP4+ neighbor.

2. Enter the **show ipv6 bgp neighbors advertised-routes** command.

```
device# show ipv6 bgp neighbors fdcd:3001:3009:1::113 advertised-routes

       There are 6 routes advertised to neighbor fdcd:3001:3009:1::113
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      fd70::/128         fdcd:3001:3009:1::118
                                          none                  0      BE
         AS_PATH: 4001183001 4031
2      fd70::1/128        fdcd:3001:3009:1::118
                                          none                  0      BE
         AS_PATH: 4001183001 4031
3      fd70::2/128        fdcd:3001:3009:1::118
                                          none                  0      BE
         AS_PATH: 4001183001 4031
4      fd70::3/128        fdcd:3001:3009:1::118
                                          none                  0      BE
         AS_PATH: 4001183001 4031
5      fd70::4/128        fdcd:3001:3009:1::118
                                          none                  0      BE
         AS_PATH: 4001183001 4031
6      fd70::5/128        fdcd:3001:3009:1::118
                                          none                  0      BE
         AS_PATH: 4001183001 4031
```

This example shows information about all the routes the BGP4+ networking device advertised to the neighbor.

3. Enter the **show ipv6 bgp neighbors last-packet-with-error** command.

```
device# show ipv6 bgp neighbors last-packet-with-error

    Total number of BGP Neighbors: 3
1   IP Address: fdcd:3001:3009:1::113
    Last error:
    BGP4: 19 bytes hex dump of packet that contains error
    ffffffff ffffffff ffffffff ffffffff 00130404
2   IP Address: fdcd:3001:3009:1::114
    Last error:
    BGP4: 19 bytes hex dump of packet that contains error
    ffffffff ffffffff ffffffff ffffffff 00130404
…
```

This example shows information about the last packet that contained an error from any of a device's neighbors.

4. Enter the **show ipv6 bgp neighbors received-routes** command.

```
device# show ipv6 bgp neighbors fd80:3001:4031:1::225 received-routes

       There are 6 received routes from neighbor fd80:3001:4031:1::225
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      fd70::/128         fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 4031
2      fd70::1/128        fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 4031
3      fd70::2/128        fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 4031
4      fd70::3/128        fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 4031
5      fd70::4/128        fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 4031
6      fd70::5/128        fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 4031
```

This example lists all route information received in route updates from BGP4+ neighbors of the device since the soft-reconfiguration feature was enabled.

5. Enter the **show ipv6 bgp neighbors rib-out-routes** command.

```
device# show ipv6 bgp neighbors fdcd:3001:3009:1::113 rib-out-routes vrf v3001

       There are 200 RIB_out routes for neighbor fdcd:3001:3009:1::113
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      fd70::/128         fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 4031
2      fd70::1/128        fd80:3001:4031:1::225
                                          none      100        0      BE
         AS_PATH: 40
```

This example shows information about BGP4+ outbound RIB routes.

Clearing BGP4+ dampened paths

BGP4+ suppressed routes can be reactivated using a CLI command.

The **show ipv6 bgp dampened-paths** command is entered to verify that there are BGP4+ dampened routes. The **clear ipv6 bgp dampening** command is entered to reactivate all suppressed BGP4+ routes. The **show ipv6 bgp dampened-paths** command is re-entered to verify that the suppressed BGP4+ routes have been reactivated.

1. Enter the **exit** command until you return to Privileged EXEC mode.

```
device(config)# exit
```

2. Enter the **show ipv6 bgp dampened-paths** command to display all BGP4+ dampened routes.

```
device# show ipv6 bgp dampened-paths

    Status Code  >:best d:damped h:history *:valid
    Network                                   From                                      Flaps
Since       Reuse       Path
*d  110:110:110:4::/64                        160:160:160::10                           36    0 :
2 :54    0 :10:10    111
*d  110:110:110:3::/64                        160:160:160::10                           36    0 :
2 :54    0 :10:10    111
*d  110:110:110:2::/64                        160:160:160::10                           36    0 :
2 :54    0 :10:10    111
*d  110:110:110:1::/64                        160:160:160::10                           36    0 :
2 :54    0 :10:10    111
*d  110:110:110::/64                          160:160:160::10                           36    0 :
2 :54    0 :10:10    111
```

3. Enter the **clear ipv6 bgp dampening** command to reactivate all suppressed BGP4+ routes.

```
device# clear ipv6 bgp dampening
```

4. Enter the **show ipv6 bgp dampened-paths** command to verify that there are no BGP4+ dampened routes.

```
device# show ipv6 bgp dampened-paths
device#
```

The following example reactivates all suppressed BGP4+ routes and verifies that there are no suppressed routes.

```
device(config-bgp-router)# exit
device(config)# exit
device# show ipv6 bgp dampened-paths
device# clear ipv6 bgp dampening
device# show ipv6 bgp dampened-paths
```

Clearing BGP4+ dampened paths

BGP4+ suppressed routes can be reactivated using a CLI command.

The **show ipv6 bgp dampened-paths** command is entered to verify that there are BGP4+ dampened routes. The **clear ipv6 bgp dampening** command is entered to reactivate all suppressed BGP4+ routes. The **show ipv6 bgp dampened-paths** command is re-entered to verify that the suppressed BGP4+ routes have been reactivated.

1. Enter the **exit** command until you return to Privileged EXEC mode.

```
device(config)# exit
```

2. Enter the **show ipv6 bgp dampened-paths** command to display all BGP4+ dampened routes.

```
device# show ipv6 bgp dampened-paths

    Status Code  >:best d:damped h:history *:valid
    Network                                   From                                      Flaps
Since       Reuse       Path
*d  110:110:110:4::/64                        160:160:160::10                           36    0 :
2 :54    0 :10:10    111
*d  110:110:110:3::/64                        160:160:160::10                           36    0 :
2 :54    0 :10:10    111
*d  110:110:110:2::/64                        160:160:160::10                           36    0 :
2 :54    0 :10:10    111
*d  110:110:110:1::/64                        160:160:160::10                           36    0 :
2 :54    0 :10:10    111
*d  110:110:110::/64                          160:160:160::10                           36    0 :
2 :54    0 :10:10    111
```

3. Enter the **clear ipv6 bgp dampening** command to reactivate all suppressed BGP4+ routes.

```
device# clear ipv6 bgp dampening
```

4. Enter the **show ipv6 bgp dampened-paths** command to verify that there are no BGP4+ dampened routes.

```
device# show ipv6 bgp dampened-paths
device#
```

The following example reactivates all suppressed BGP4+ routes and verifies that there are no suppressed routes.

```
device(config-bgp-router)# exit
device(config)# exit
device# show ipv6 bgp dampened-paths
device# clear ipv6 bgp dampening
device# show ipv6 bgp dampened-paths
```

BGP EVPN for IP Fabrics

- Overview....439
- BGP EVPN control plane....442
- BGP EVPN configuration examples....443
- BGP routing tables....449
- BGP EVPN-based MCT cluster formation....450
- BGP EVPN-based VXLAN overlay....453
- BGP EVPN in IPv6-only underlay....458
- BGP EVPN in an MPLS provider network....458
- EVPN next-hop resolution and tunnel-EVI membership....459
- BGP EVPN in an MPLS provider network....460
- Layer 2 (MAC) route exchange....462
- ARP/ND (MACIP) route exchange....466
- EVPN prefix route exchange and Multi-VRF support....467
- BGP EVPN VXLAN data center interconnect....471
- Static anycast gateway....473
- IP unnumbered interface....475
- High availability....476

Overview

A multiprotocol extension for BGP to carry Layer 2 routes was formally standardized in 2015 with RFC 7432. Even though EVPN was initially targeted as WAN technology for data-center interconnect, replacing VPLS, gradual extensions to the base standard have helped it also gain interest within the data center. EVPN provides a standards-based solution for overlay interconnect within a data center.

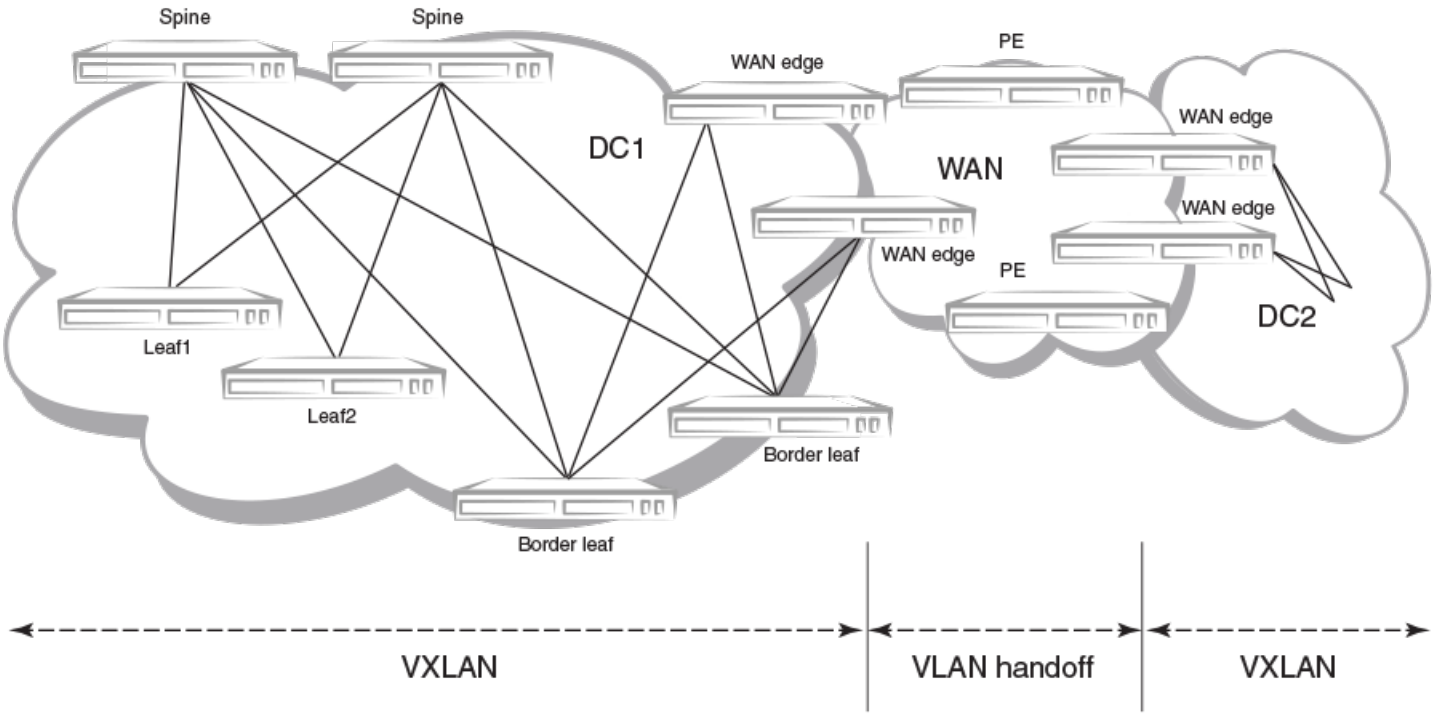
BGP EVPN for VXLAN overlay networks, along with a broad set of Layer 2, Layer 3, and Infrastructure features to enable the seamless deployment, on-demand usage of forwarding entries in hardware, and minimization of flooding in the network, are referred to as a BGP EVPN-based IP Fabric. This feature is available in VDX platforms series starting with Network OS 7.0.0. Limited BGP EVPN functionality, for spines and leaf nodes (Layer 2 gateways), is supported beginning with SLX-OS17r.1.00.

Most of the contents in this chapter are common for data center overlay and data center interconnect (DCI).

IP Fabrics topologies

RFC 7432 ("BGP MPLS-Based Ethernet VPN") specifies multiprotocol extensions to BGP to exchange Layer 2 routes in an MPLS Ethernet VPN network. These extensions can be useful in DCI scenarios. The following figure shows a supported topology where the SLX 9850 or SLX 9850 act as nodes in a BGP EVPN spine in an IP Fabrics network.

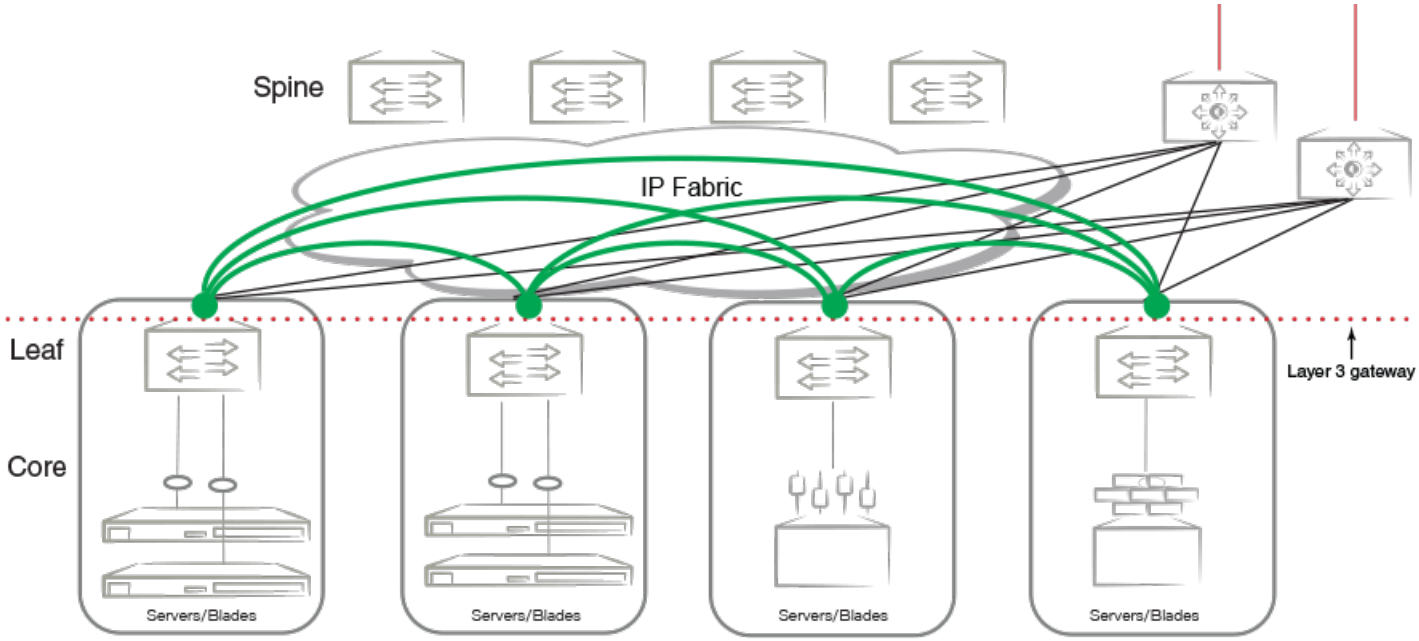
FIGURE 49 Supported BGP EVPN spine topology



The leaf nodes in the IP Fabric are VDX switches. A BGP EVPN-based Multi-Chassis Trunk (MCT) solution continues to be supported on those platforms. However, using a colocated BGP EVPN spine and a BGP EVPN MCT on the same switch is not supported.

VXLANs provide the interconnectivity between leaf nodes in an IP Fabric network. VXLAN tunnels are established between leaf nodes. The following figure illustrates this VXLAN connectivity.

FIGURE 50 VXLAN connectivity in an IP Fabric



IP Fabrics features

This release supports the following IP Fabrics features:

- BGP EVPN control plane
- VXLAN VTEP discovery
- BGP EVPN-based MCT cluster formation
- Layer 2 (MAC) route exchange
- ARP/ND (MACIP) route exchange (with MAC-IP support only in the default VRF)
- Static anycast gateway
- Conversational MAC learning
- Conversational ARP learning
- ARP suppression
- IP unnumbered interfaces (minimal support for tunnel next-hop; no routing protocol support)
- Multi-VRF support and EVPN prefix route exchange
- Conversion of MACIP routes to host prefix routes
- BGP dynamic neighbors

  NOTE
  Layer 3 over port-channel is not supported.

BGP EVPN use cases

The following BGP EVPN use cases are addressed:

- BGP EVPN VXLAN overlay within the data center
- MCT cluster formation
- VXLAN data-center interconnect
- EVPN MPLS data-center interconnect
- EVPN MPLS provider network support

BGP EVPN control plane

This section discusses the following components of the BGP EVPN control plane.

Supported route types

All EVPN route types described in RFC7432 are supported, including the Type-5 IP prefix route described in the IETF draft, "draft-ietf-bess-evpn-prefix-advertisement." The following table describes these types.

TABLE 26 Supported EVPN route types

Type	Description
1	Ethernet Auto Discovery (AD) route
2	MAC/IP advertisement route
3	Inclusive Multicast Route
4	Ethernet Segment Route
5	IP Prefix Route

Supported BGP functionalities

The following features are supported for BGP EVPN address-family:

- iBGP and eBGP peering
- IPv4 and IPv6 peer types
- Peer groups with EVPN neighbors
- Ability to negotiate only EVPN address-family for IPv4/IPv6 peers
- iBGP route-reflector server
- Full iBGP mesh without route-reflecter server
- Keeping next-hop unchanged for EBGP peers
- Ability to retain all EVPN routes without importing them
- Coexistence of all IPv4 and IPv6 BGP features for neighbors negotiating EVPN address-family
- EVPN spine, leaf, and border leaf functionalities

Supported data plane encapsulation types

The following data-plane encapsulation types are supported:

-
- VXLAN

MPLS is the default encapsulation type.

The BGP encapsulation extended-community attribute is carried with each route in the BGP control plane to signify the encapsulation to be used to reach the prefix. An EVPN route not carrying any encapsulation extended-community attribute implicitly means that MPLS is encapsulation type.

If an EVPN prefix is received from multiple sources with both VXLAN and MPLS encapsulations, the route source with VXLAN encapsulation is preferred within the data center.

Supported service-interface model

The VLAN-based service-interface model described in RFC 7432 is supported. Each VLAN is mapped to a unique VNI/MPLS label. Within the data center, each VNI maps to a unique EVI; this mapping is referred to as the "single subnet per EVI" option in "draft-ietf-bess-evpn-overlay." No VLAN translation service is supported. Within the data center, VNI-to-VLAN mapping is local to each leaf, and the inner VXLAN frames may not carry an Ethernet tag. The administrator is responsible for keeping this mapping consistent within the data center.

BGP EVPN configuration examples

BGP EVPN neighbor examples

This section presents configuration examples of basic BGP EVPN neighbor configurations.

Configuring a BGP EVPN neighbor and peer group

EVPN address-family can be configured to be negotiated on either IPv4 or IPv6 BGP neighbors, as shown in the following example.

```
device(config)#router bgp
device(config-bgp-router)# local-as 62001
device(config-bgp-router)# neighbor 192.168.21.31 remote-as 62000
device(config-bgp-router)# neighbor 2000:10:1::2 remote-as 62002
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# retain route-target all <---Spine only
device(config-bgp-evpn)# neighbor 192.168.21.31 next-hop-unchanged <---Spine only
device(config-bgp-evpn)# neighbor 192.168.21.31 activate
device(config-bgp-evpn)# neighbor 2000:10:1::2 next-hop-unchanged <---Spine only
device(config-bgp-evpn)# neighbor 2000:10:1::2 activate
device(config-bgp-evpn)# end

device(config)#router bgp
device(config-bgp-router)# local-as 62001
device(config-bgp-router)# neighbor 192.168.21.31 remote-as 62000
device(config-bgp-router)# neighbor 2000:10:1::2 remote-as 62002
device(config-bgp-router)# address-family evpn
device(config-bgp-evpn)# retain route-target all <---Spine only
device(config-bgp-evpn)# neighbor 192.168.21.31 next-hop-unchanged <---Spine only
device(config-bgp-evpn)# neighbor 192.168.21.31 activate
device(config-bgp-evpn)# neighbor 2000:10:1::2 next-hop-unchanged <---Spine only
device(config-bgp-evpn)# neighbor 2000:10:1::2 activate
device(config-bgp-evpn)# end
```

As with any VPN technology, EVPN neighbors must always be in the default VRF. The preceding example shows EVPN address-family being activated for IPv4 and IPv6 BGP neighbors.

BGP neighbor configuration can be simplified by using peer groups, as in the following example.

```
device(config-bgp-router)# neighbor my-peer-group peer-group
device(config-bgp-router)# neighbor my-peer-group remote-as 62001
device(config-bgp-router)# neighbor 192.168.30.2 peer-group my-peer-group
device(config-bgp-router)# neighbor 2000:20:1::2 peer-group my-peer-group
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# retain route-target all
device(config-bgp-evpn)# neighbor my-peer-group route-reflector-client
device(config-bgp-evpn)# neighbor my-peer-group activate
```

```
device(config-bgp-router)# neighbor my-peer-group peer-group
device(config-bgp-router)# neighbor my-peer-group remote-as 62001
device(config-bgp-router)# neighbor 192.168.30.2 peer-group my-peer-group
device(config-bgp-router)# neighbor 2000:20:1::2 peer-group my-peer-group
device(config-bgp-router)# address-family evpn
device(config-bgp-evpn)# retain route-target all
device(config-bgp-evpn)# neighbor my-peer-group route-reflector-client
device(config-bgp-evpn)# neighbor my-peer-group activate
```

NOTE
The **neighbor route-reflector client** command is not required if all devices peer with each other in a full iBGP mesh.

Specifying BGP EVPN neighbor encapsulation

BGP EVPN routes carry the encapsulation type as part of the BGP encapsulation extended community. When BGP encapsulation extended community is absent from the route, the default MPLS encapsulation is assumed. On SLX-OS platforms, the default encapsulation for EVPN peers is MPLS, and BGP encapsulation extended community with the MPLS encapsulation type is attached to all EVPN routes.

On SLX-OS platforms, the user should configure encapsulation for each BGP EVPN neighbor before activating it under EVPN address-family.

Encapsulation for BGP EVPN peers can be specified as in the following example.

```
device(config-bgp-router)# address-family evpn
device(config-bgp-evpn)# neighbor 192.168.30.2 encapsulation vxlan
device(config-bgp-evpn)# neighbor 2000:20:1::2 encapsulation vxlan
device(config-bgp-evpn)# neighbor my-peer-group encapsulation vxlan
```

Retaining EVPN routes without importing them

For spine or specific border leaf cases, routes are not imported into MAC-VRF or IP-VRF tables, but instead are reflected/re-advertised to iBGP/eBGP peers. The following configuration is required to retain all of the routes in the VPN table.

```
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# retain route-target all
```

```
device(config-bgp-router)# address-family evpn
device(config-bgp-evpn)# retain route-target all
```

Keeping next-hop unchanged for eBGP neighbors

When BGP advertises routes to an eBGP peer, the next-hop value in the route is modified to carry the local peering IP address. In the case of a BGP EVPN VXLAN IP Fabric, the next-hop value in the route is the VTEP IP address of the advertising router. A BGP EVPN

spine distributing the routes to other leaf nodes should not modify the next-hop value. Each EBGP peer activated under EVPN address-family should be configured so that it does not modify the next-hop value, as in the following example.

```
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 next-hop-unchanged
device(config-bgp-evpn)# neighbor my-peer-group next-hop-unchanged
device(config-bgp-evpn)# neighbor 192.168.21.31 activate
device(config-bgp-evpn)# neighbor my-peer-group activate

device(config-bgp-router)# address-family evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 next-hop-unchanged
device(config-bgp-evpn)# neighbor my-peer-group next-hop-unchanged
device(config-bgp-evpn)# neighbor 192.168.21.31 activate
device(config-bgp-evpn)# neighbor my-peer-group activate
```

Allowing routes with a local AS number

In special cases where spine or super-spine routers must accept EVPN routes that have the AS number of a router in the AS path segment, the **allowas-in** command can be used for a given BGP peer, as in the following example.

```
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 allowas-in 1
device(config-bgp-evpn)# neighbor my-peer-group allowas-in 1

device(config-bgp-router)# address-family evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 allowas-in 1
device(config-bgp-evpn)# neighbor my-peer-group allowas-in 1
```

Configuring iBGP EVPN neighbors

In a full iBGP mesh of EVPN nodes, a leaf node needs to peer only with all other leaf nodes; it does not need to have a session with any spine node. The configuration of route-reflector client on the spine nodes is not required.

In a non-full iBGP mesh configuration, in order to reflect routes from one iBGP neighbor to another iBGP neighbor, the iBGP neighbors should be configured as route-reflector-clients, as in the following example.

```
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 route-reflector-client
device(config-bgp-evpn)# neighbor my-peer-group route-reflector-client

device(config-bgp-router)# address-family evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 route-reflector-client
device(config-bgp-evpn)# neighbor my-peer-group route-reflector-client
```

The '"client-to-client-reflection" feature is by default enabled under EVPN address-family. The route-reflector configuration does not force routes that are not imported to be retained in BGP. The **retain route-target all** command is required in all cases.

Filtering EVPN routes

Limited filtering support is available for EVPN routes. However, route maps can be configured on EVPN neighbors, as in the following example.

```
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 route-map in rmap-in
device(config-bgp-evpn)# neighbor 192.168.21.31 route-map out rmap-out

device(config-bgp-router)# address-family evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 route-map in rmap-in
device(config-bgp-evpn)# neighbor 192.168.21.31 route-map out rmap-out
```

Only AS path and extended-community-based match and set criteria are supported for EVPN routes.

Negotiating only EVPN address-family

In order to have better resiliency, separate BGP neighbors for overlay and underlay may be desired. This separation allows administrators to manipulate, debug, and maintain EVPN neighbors without affecting underlay connectivity. This separation can be achieved by deactivating underlay address-family for EVPN neighbors.

IPv4 neighbors are automatically activated under IPv4 unicast address-family, and they must be deactivated as in the following example.

```
device(config-bgp-router)# neighbor 192.168.21.31 remote-as 62000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# no neighbor 192.168.21.31 activate
device(config-bgp-ipv4u)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 activate

device(config-bgp-router)# neighbor 192.168.21.31 remote-as 62000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# no neighbor 192.168.21.31 activate
device(config-bgp-ipv4u)# address-family evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 activate
```

IPv6 neighbors are not by default activated under IPv6 unicast address-family. Therefore, for them to negotiate only EVPN address-family, they must be activated only under EVPN, as in the following example.

```
device(config-bgp-router)# neighbor 2000:10:1::2 remote-as 62002
device(config-bgp-ipv4u)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 2000:10:1::2 activate

device(config-bgp-router)# neighbor 2000:10:1::2 remote-as 62002
device(config-bgp-ipv4u)# address-family evpn
device(config-bgp-evpn)# neighbor 2000:10:1::2 activate
```

Enabling peer AS check

When a BGP router receives an update message with its own AS number in the AS path attribute, this means that there is an AS path loop and by default such updates are discarded. In many scenarios this AS path check needs to be disabled, and seeing a the AS number of a router in the AS path attribute is fine. On SLX devices, this AS path check is enforced at the receiver and is overridden by means of the **neighbor allowas-in** command. However, enforcing this AS path check at the sender side can save the amount of RIBout memory used to store the routes that are discarded at the receiver. Furthermore, it avoids sending updates and withdraws those routes, thereby improving the convergence time.

The following example configuration shows how to enforce this check for a neighbor or peer group. This configuration is per address-family.

```
LeafC_2(config-bgp-router)# address-family ipv4 unicast
LeafC_2(config-bgp-ipv4u)# neighbor 163.124.20.10 enable-peer-as-check
```

A neighbor reset is required for the preceding configuration to take effect. After it is applied, this configuration prevents Network Layer Reachability Information (NLRI) from being added to the RIBout of the peer if the AS path segment of the NLRI has the AS number configured as "remote-as" for the neighbor.

BGP EVPN instance configuration

With a VLAN-based service model, each VLAN/BD corresponds to a unique EVPN instance, or MAC-VRF. Each route in EVPN belonging to a MAC-VRF has a unique route distinguisher (RD) that differentiates routes from other MAC-VRFs.

In addition, sets of route targets (export RTs) are associated with each MAC-VRF; these RTs are applied to the routes while advertising. RTs in the route are matched against the configured import RTs. If any of the RTs match, the route is imported; otherwise, it is discarded.

NOTE
Auto RD is mandatory for EVI extension with manual RD and RT/auto RT.

Each VLAN/BD to be extended into an EVPN overlay must be added under the EVPN configuration block, as shown in the following configuration example.

```
evpn t1
!
vlan 1020
  rd 1020:1
  route-target both 1020:100
!
vlan 3200
 rd 3200:1
 route-target both 3200:100
!
bridge-domain 10
 rd 10:1
 route-target both 10:100
!
bridge-domain 3100
 rd 3100:1
 route-target both 3100:1
```

Each VLAN/BD added to the preceding EVPN configuration is considered an EVI and is assigned a unique EVID internally. EVIDs for VLANs/BDs are calculated as shown in the following table.

TABLE 27 Calculating EVIDs for VLANBs/BDs

VLAN/BD	EVI value
VLAN (1–4095)	VLANID
BD (1–4096+)	4096 + BD-ID

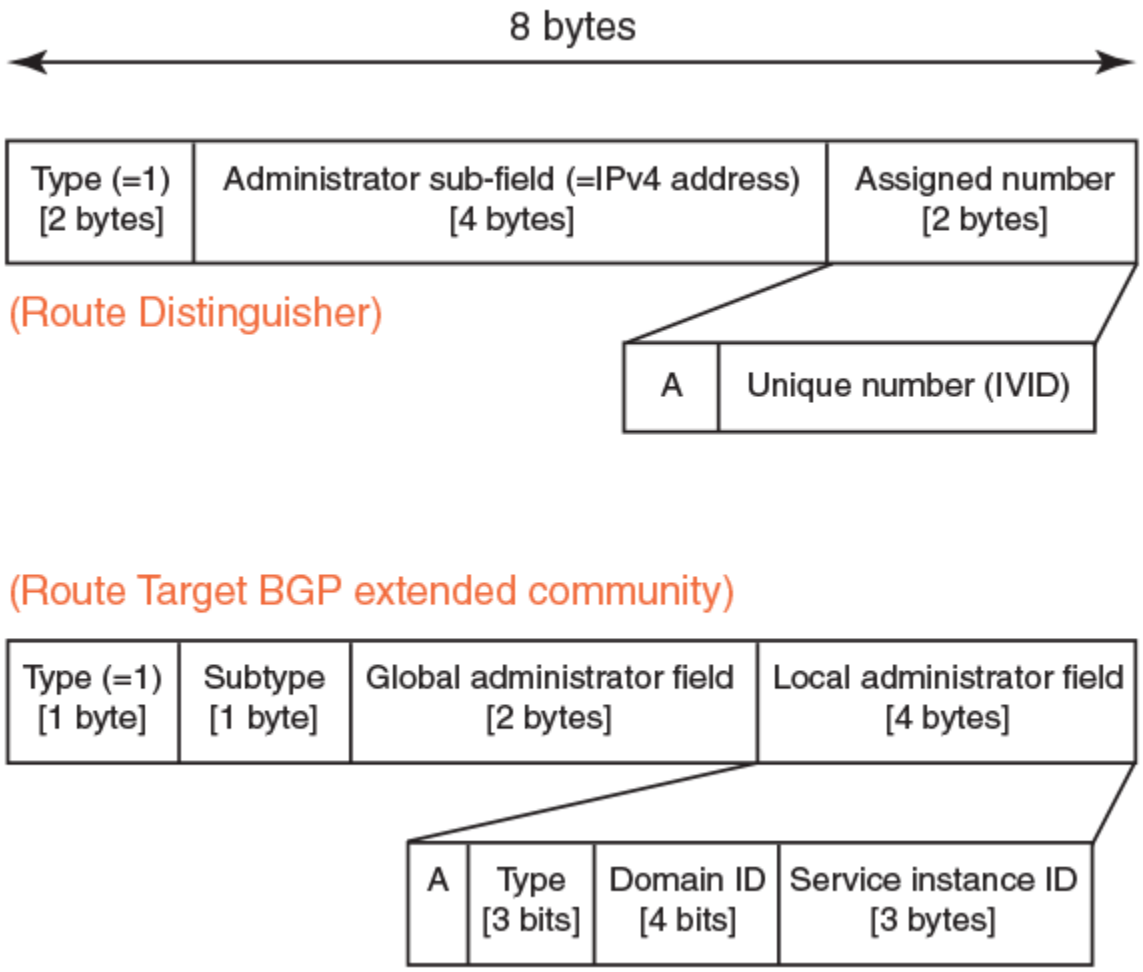
The EVID value in the preceding configuration example is not sent in the Ethernet-tag field in the routes. It is always 0 for VLAN-aware service. However, routes imported into the MAC-VRF table are classified against an EVID, which is shown in the Ethernet-tag field.

Autogeneration of route distinguisher and route target

Because thousands of VLANs/BDs can exist in the network, configuring each of them individually on each router is cumbersome. In order to simplify the configuration, Extreme recommends the autogeneration of RDs and RTs.

The RD of EVPN routes is encoded as a Type-1 route-distinguisher, as described in RFC4364. The following figure shows the formats of RD and BGP RT extended community.

FIGURE 51 RD and RT formats



The most significant bit "A" of the assigned number field stands for automatic or manual RD. For a manual RD value, this bit is set and the unique identifier value is set to the internal VLAN ID (IVID) to keep the RD unique across VLANs and bridge domain IDs (BD-IDs). The administrator subfield of the RD is set to the BGP router ID.

The global administrator field in the RT is set to the AS number. The local administrator field in the RT is expanded, as shown in the preceding figure. Bit-value "A" stands for automatic or manual RT. Where the RT value is derived automatically, this bit is set to 0. The service instance ID field carries the VLAN or VNI value. The Type field is set, according to the value set in the service instance ID space, to either 0 (= IVID) or 1 (= VNI). The domain-id value is not used currently and is set to 0.

RTs with the following service instance types are attached according to neighbor encapsulation:

TABLE 28 Neighbor encapsulation and RT service instance types

Neighbor encapsulation	RT service instance type
VXLAN	VXLAN (1)
MPLS	EVID (4)

The configuration of VLANs/BDs that use autogenerated RD and RT values is simplified, as shown in the following configuration example.

```
evpn
 rd auto
 route-target both auto ignore-as
 vlan add 1000-2000
 bridge-domain add 1000-2000
```

VLAN and bridge domains that are added with range, RD, and RT values are specified as "auto". In scenarios where multiple leaf nodes belong to different autonomous systems, Extreme recommends the "ignore-as" option. When that option is specified, the AS number field in the RT in the route is not compared against the local AS number.

BGP routing tables

The EVPN table holds VPN routes belonging to the L2VPN EVPN address-family. These routes may be received from an EVPN peer, originated locally, or exported from other VRFs for prefix routes.

The EVPN table holds VPN routes belonging to the EVPN address-family. These routes may be received from an EVPN peer, originated locally, or exported from other VRFs for prefix routes.

EVPN route types 1 through 4 received from remote peers are imported only into the MAC-VRF table, except for MACIP host routes that may get converted to IPv4/v6 host routes and imported into the IP-VRF table. EVPN routes received from remote peers are validated against import rules and are added to the VPN table only if validation passes. The following table lists the import rules for peer types and route types.

TABLE 29 Import rules for peer and route types

Received from peer type	Route type	Import rules
VXLAN peer	MAC/IP, Ethernet Tag	RTs in the route are looked up in the MAC-VRF table. A route is imported for the following conditions: 1. RT in the route matches RTs configured for VLAN/BD. 2. When a VLAN/BD instance is not configured manually, a VLAN/BD must exist in the system and be added to EVPN. 3. L2VNI in the label field is same as corresponding VNI for the VLAN/BD. For MAC-IP (ARP/ND) routes, if no match is found in MAC-VRF, RTs in the route are looked up in IP-VRFs. For each IP-VRF matching the RT, the route is imported in that IP-VRF.
	IP Prefix	RTs in the route are looked up in IP-VRFs. For each IP-VRF matching the RT, route is imported into that IP-VRF.
	Ethernet Segment	Route is imported if the RT in the route matches the auto ES-RT of the locally configured MCT interface.
	Inclusive Multicast Routes	

The EVPN table can hold routes even if they are not imported into MAC-VRF or IP-VRF tables. This table is required at the spine or border-leaf nodes to reflect the routes to other leaf nodes or to other data centers, respectively. The **retain route-target all** command is used to configure this table.

The consolidation of the routes from different sources (RDs) occurs in the MAC-VRF table after the routes pass through route-target checking and import filtering. The following operations are performed exclusively in the MAC-VRF table:

- VTEP discovery
- MAC move detection
- MAC dampening
- ES-based route use

The following table lists advertisement behavior by route type.

TABLE 30 Advertisement behavior by route type

Route type	Advertised to VXLAN peer?	Advertised to MPLS peer?
Autodiscovery	No	Yes
MAC	Yes (without ESI)	Yes
MACIP (ARP/ND)	Yes (without ESI)	Yes
Inclusive Multicast	Yes	Yes
Ethernet Segment (ES)	No	Yes
IP-Prefix (IPv4Prefix/IPv6Prefix)	Yes	Yes (without router MAC)

TABLE 31 Advertisement behavior by route type

Route type	Advertised to VXLAN peer?
Autodiscovery	No
MAC	Yes (without ESI)
MACIP (ARP/ND)	Yes (without ESI)
Inclusive Multicast	Yes
Ethernet Segment (ES)	No
IP-Prefix (IPv4Prefix/IPv6Prefix)	Yes

BGP EVPN-based MCT cluster formation

Multi-Chassis Trunking (MCT) cluster formation leverages BGP EVPN Ethernet Segment and Auto-Discovery (AD) routes and employs proprietary mechanisms to avoid loops in the cluster in transient configuration scenarios.

No separate EVPN instance for MCT and non-MCT services exists. Therefore, in order to configure MCT even on a stand-alone MCT cluster, EVPN configuration is required.

Dependency on the EVPN instance

EVPN configuration is required (along with BGP neighbor and MCT cluster/client) to form an MCT cluster. The MCT cluster automatically becomes a member of the VLAN/BDs configured under EVPN. The following is an example configuration.

```
cluster mct-cluster 1
  peer-interface ethernet 0/1
  peer 110.1.1.2
  deploy
```

```
    client mct1 1
      client-interface Port-channel 10
      esi 00:00:11:22:33:44:55:66:77
      deploy
  !

  evpn
    rd auto
    route-target both auto ignore-as
    vlan add 100,200,1000-4000
    bridge-domain add 100,200,1000-4000
```

The above configuration creates an EVPN configuration instance with auto RD and RT and specified VLAN/BDs being added to the EVPN configuration.

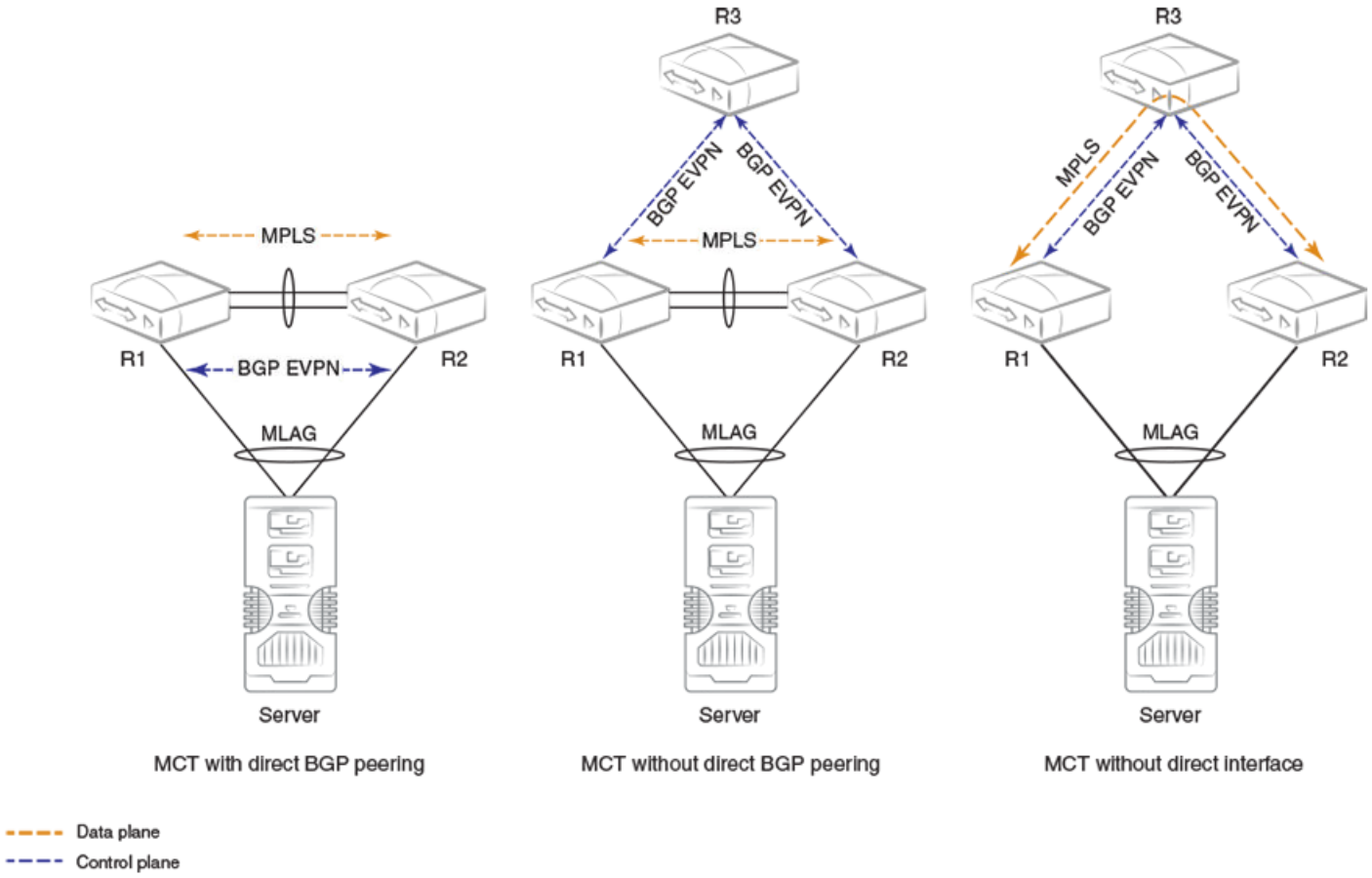
BGP EVPN peering for the MCT cluster

NOTE

MCT Peers need to run the same version of SLX-OS.

MCT cluster formation is controlled by BGP EVPN peering between the cluster members, and it should be configured with the MPLS encapsulation type. The following figure illustrates typical MCT use cases.

FIGURE 52 MCT use cases



BGP EVPN peering (not necessarily direct) between the MCT cluster members is required.

The first diagram shows a standalone MCT cluster and is the simplest MCT use case, where BGP EVPN peering exists only between the cluster members. An MPLS tunnel is established over the direct link (peer-interface), and the control and data planes are established over the same path.

The second diagram shows separate control and data plane paths. BGP EVPN peering is not direct between the cluster members.

The third diagram shows MCT cluster formation without a direct link (peer interface). In this case, a multihop MPLS tunnel is established through intermediate routers.

The MCT cluster state is no longer dependent on BGP peering with the peer IP address specified under cluster configuration. Instead, the reachability of the BGP EVPN next-hop of the corresponding encapsulation is used for the cluster state.

Handling of ES/AD routes in BGP

ES and AD-per-ES routes are exchanged between MCT peers for Designated Forwarder (DF) election. The ES route is originated when the MCT cluster client is deployed. A set of AD-per-ES routes is originated when the cluster client interface is operationally up.

ES routes carry BGP ES-import route-target extended community, which is automatically constructed from the ESI value. ES-import RTs are automatically added in EVPN whenever a cluster client interface with a given ES is configured. An ES route with a given ESI value is imported only by those routers that have membership of that ES.

Similar to other EVPN routes, ES/AD routes are added to BGP RIB-in and are advertised to EVPN neighbors.

AD-per-ES routes are required to carry the route-targets of all EVIs that are members of the ES (cluster client interface). Because a large number of EVIs can be members of the ES (4K VLAN + BDs), the AD route may not fit in a single BGP update message, which is only 4K. Multiple AD-per-ES routes with unique RDs are generated for each ES as shown in the following table.

TABLE 32 RD index and EVI RTs

RD index for RD (Router-ID: RD-index)	RTs of the EVIs attached to AD-per-ES route
1	VLAN 1-127
2	VLAN 128-255
3	VLAN 256-383
. . .	. . .
32	VLAN 3968-4095
33	BD 1-127
34	BD 128-255
35	BD 256-383
...	...

AD-per-ES routes carry the ESI label in ESI label extended community for split-horizon filtering.

BGP EVPN-based VXLAN overlay

BGP EVPN provides control plane signaling and tunnel discovery support in a VXLAN overlay network.

Inclusive multicast routes provide for the extension of VLANs/BDs over tunnels, and MAC, MACIP, and prefix routes provide for the signaling of control plane routes for tenants. Forwarding across the tenants or toward the core takes place through VXLAN overlay tunnels. The flooding of BUM traffic is achieved by means of ingress replication.

The following points apply to a BGP EVPN-based VXLAN overlay network:

- A mesh of tunnels from everywhere to everywhere is required within the data center.
- Flooding is achieved through Ingress replication; multicast support does not exist.
- Spine nodes provide load balancing for overlay tunnels and alternate route sources for BGP.
- LVTEP provides inherent load balancing in the underlay for dual-homed hosts. There is no need for "aliasing" as specified in RFC 7432.

Devices in the underlay Layer 3 Clos network may have the following distinct roles:

- Leaf
- Spine
- Border leaf
- Colocated border leaf and spine

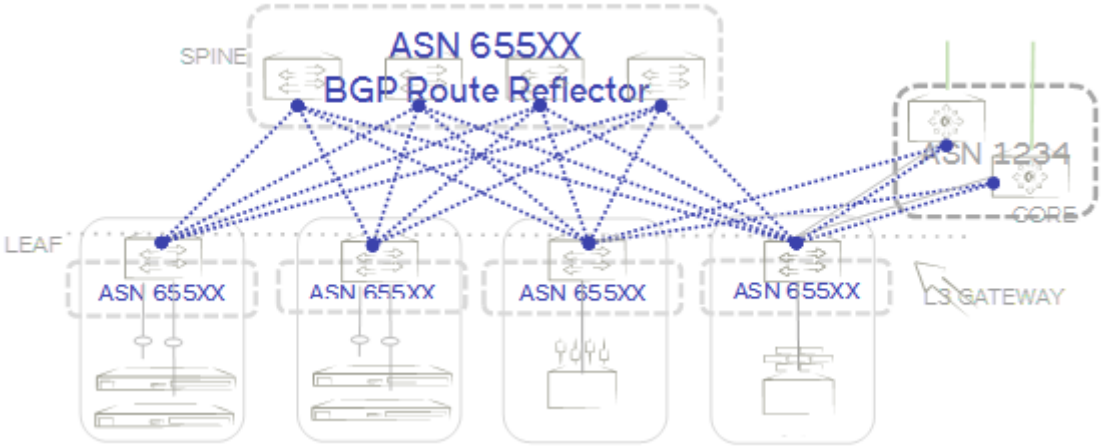
Underlay architectures

The underlay architecture in a Layer 3 Clos network that can be based on iBGP or eBGP. The following topologies illustrate these options.

iBGP spine and leaf

With iBGP-based underlay, all spine and leaf nodes are in the same AS. Spine nodes act as route-reflectors and do not terminate any tunnel, whereas leaf nodes originate and terminate all of the tunnels and routes. The following figure illustrates this topology.

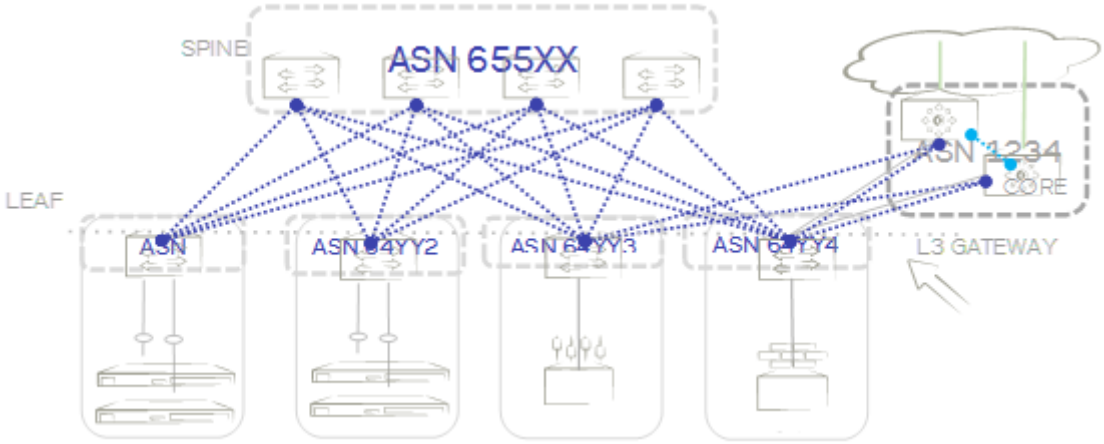
FIGURE 53 iBGP spine and leaf



eBGP spine and leaf

With eBGP-based underlay, each leaf node is assigned a distinct AS number. Usually, the iBGP underlay architecture provides higher BGP scalability but with less control on the routes getting exchanged. On the other hand, eBGP underlay provides lower BGP scalability but with greater control on routes, as policy and filters can be applied on the originating AS numbers. The following figure illustrates this topology.

FIGURE 54 eBGP spine and leaf

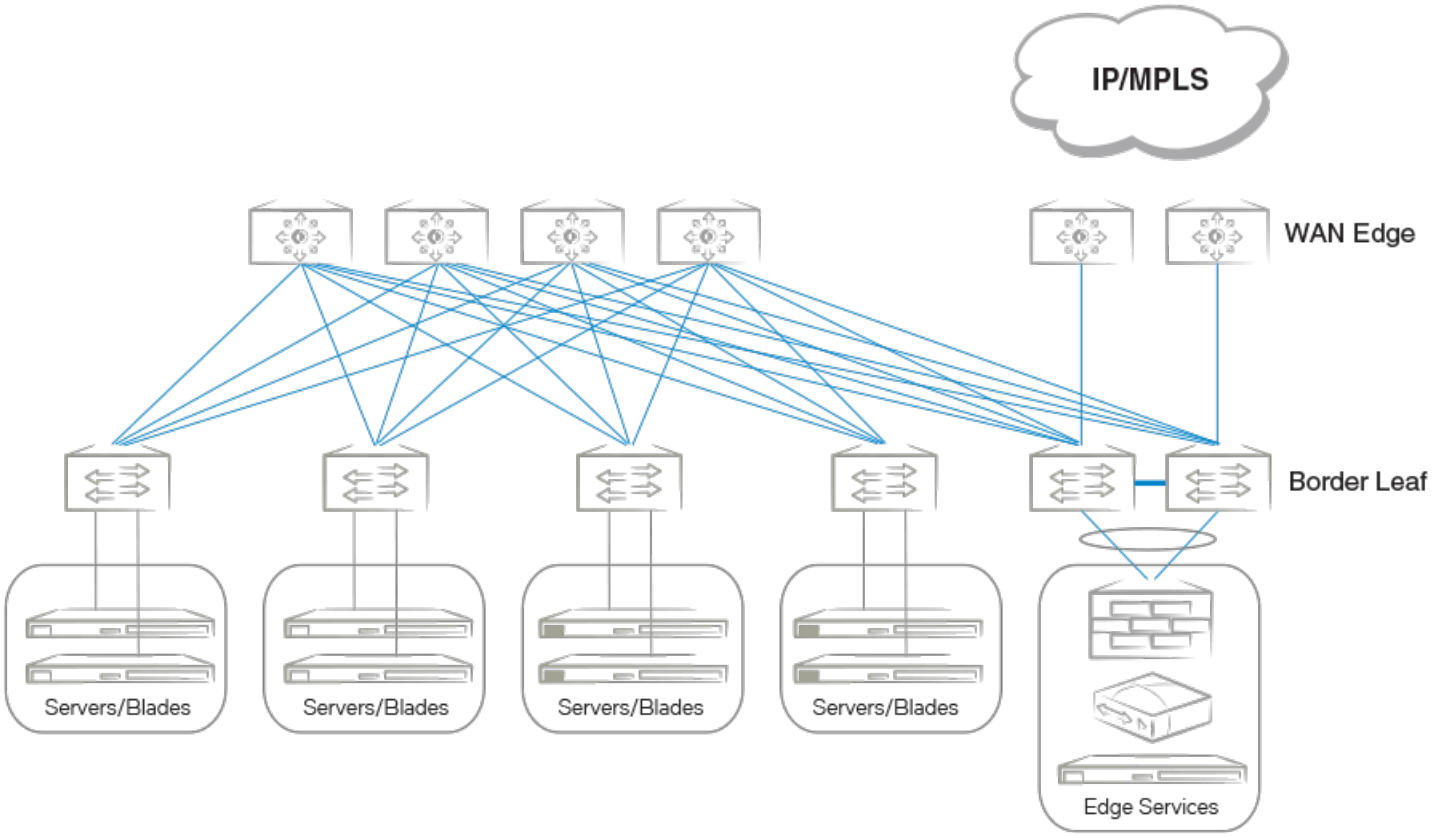


Border and service leaf

A border leaf is special leaf node, typically redundant, that sits at the edge of the data center and provides termination, handoff, or control-plane extension towards the WAN or core. Similarly, that functionality is provided in the reverse direction, from the WAN/core towards the data center.

A border leaf may act as a transparent route-reflector/forwarder (iBGP/eBGP) or may terminate tunnels while providing reflection service. In the service-leaf model, a border leaf also acts as just another leaf node in the network and may be extending Layer 2, Layer 3, or both services as illustrated in the following figure.

FIGURE 55 Border and service leaf



Colocated spine and border leaf

With a Layer 2/Layer 3 control-plane extension model, a border leaf may provide just a route forwarding service—which is merely spine functionality. In the absence of a separate border leaf, spine nodes can themselves act as route reflectors towards the WAN/core.

VXLAN overlay configuration

An overlay gateway configuration must be present and should remain in the active state in order to exchange routes in the VXLAN overlay network. An example configuration follows.

```
overlay-gateway gateway1
  type layer2-extension
  ip interface Loopback 1
  map vni auto
  activate
!

interface Loopback 1
  ip address 192.168.32.10/32
  no shutdown
!
```

The following are criteria for accepting or originating routes in a VXLAN overlay network:

- An overlay gateway is configured.
- The overlay gateway is activated.

- The tunnel type is Layer 2 extension.
- The source VTEP IP address is configured. In case the source IP address is obtained from a loopback interface, the loopback interface must be up.
- VLAN/BD-to-VNI mapping (auto or manual) exists to accept and originate routes for a specific VLAN/BD.

The preceding criteria affect only the exchange of routes between BGP neighbors with VXLAN encapsulation. When the overlay gateway is deactivated or unconfigured, all dynamic tunnels are deleted, EVPN routes received with VXLAN encapsulation are removed, and the RIBout of the BGP EVPN neighbors configured with VXLAN encapsulation is flushed.

Dynamic VTEP discovery

Dynamic VTEP discovery is a configurable option. It can be enabled under BGP EVPN address-family configuration mode, as shown here.

> **NOTE**
> This feature is supported on static VXLAN tunnels with non-MCT nodes, for regular VTEPs but not for logical VTEPs.

```
router-bgp
  address-family l2vpn evpn
    vtep-discovery
!

router-bgp
  address-family evpn
    vtep-discovery
!
```

By default VTEP discovery is enabled. When enabled, if a route with BGP encapsulation extended community as VXLAN type is imported into the MAC-VRF table, a VXLAN tunnel (if it does not already exist) is created in the system with the destination IP address as the next-hop IP address of the route. MAC learning in the forwarding plane on BGP-discovered VXLAN tunnels is automatically disabled.

When the preceding "vtep-discovery" is disabled, all dynamically discovered VXLAN tunnels are deleted from the system. VTEPs can be administratively configured, and they can be instructed to disable MAC learning in the forwarding plane and instead to use BGP, as in the following example.

```
overlay-gateway gateway1
  type layer2-extension
...
  site tunnel1
    ip address 192.168.32.20
    extend vlan add 10-100
    extend bridge-domain add 10-100
!
  activate
!
```

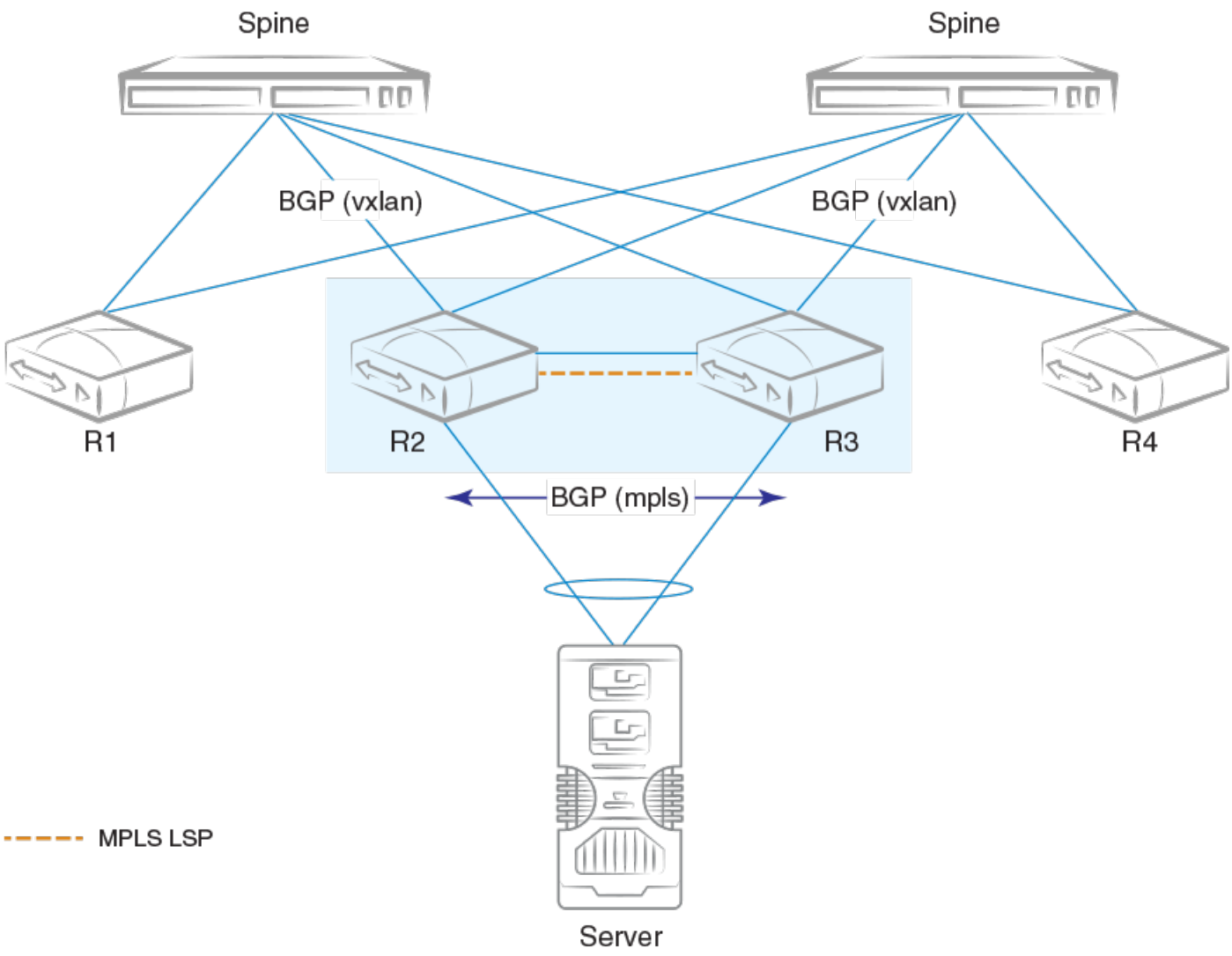
VLANs/BDs on static tunnels are not extended by BGP over inclusive-multicast routes received from remote nodes. They should be extended manually, as shown in the preceding configuration. Only when specific VLANs/BDs are extended does BGP download the corresponding EVPN routes pointing to the appropriate tunnel. A statically configured VXLAN tunnel overrides any dynamically discovered tunnel. Similarly, when a static tunnel is removed, a dynamic VTEP is created if a route with that next-hop IP address is present in the BGP MAC-VRF table.

MCT and logical VTEP

MCT cluster configuration is required for logical VTEP (LVTEP) operation. MCT peers should have BGP EVPN route reachability with the corresponding encapsulation (MPLS).

The following figure shows a typical example of a two-node MCT in a VXLAN IP Fabric. MCT cluster members have two BGP peerings: one with MCT cluster encapsulation (MPLS) and the other with VXLAN encapsulation to the spine.

FIGURE 56 MCT with LVTEP and direct BGP peering



In the presence of an LVTEP, BGP with VXLAN encapsulation cannot be used to exchange EVPN routes between MCT peers, because (1) VXLAN routes are originated with a source VTEP IP address that is same across MCT peers, and (2) VXLAN as a transport between MCT peers is not supported.

The following are requirements or recommendations for overlay gateway configuration on MCT cluster nodes:

- Configuring MCT cluster nodes as individual VXLAN tunnel end points is not recommended. MCT cluster nodes should be configured to form an LVTEP if VXLAN overlay is required.
- The same source VTEP IP address should be configured on all nodes in an MCT cluster to form an LVTEP.
- The same set of VLANs/BDs should be configured in the system and added under EVPN on all nodes in an MCT cluster.
- VLAN/BD-to-VNI mapping should be the same on all nodes in an MCT cluster.

- If routing is enabled on a VLAN/BD, static anycast gateway is recommended.

BGP EVPN in IPv6-only underlay

As described in previous sections, BGP EVPN address-family can be negotiated over both IPv4 and IPv6 BGP peers.

In case EVPN BGP peering uses IPv6, the next-hop in the EVPN MP_REACH_NLRI is still encoded as a four-byte address, and routes are installed on the remote router pointing to an IPv4 next-hop. When the underlay network is purely IPv6, resolving IPv4 next-hop becomes challenging and requires IGP or BGP for IPv4 address-family in the underlay network in order to exchange routes for these IPv4 next-hops. In a VXLAN network, these next-hop addresses are merely IPv4 VTEP IP addresses.

RFC5549 specifies an extension to Multiprotocol BGP (MP-BGP) for encoding IPv4 routes with IPv6 next-hop. An IPv6 BGP peer can be configured to exchange "Extended Next Hop Encoding" capability if it is activated under IPv4 address-family, as shown in the following example.

```
device(config)#router bgp
device(config-bgp-router)# local-as 62001
device(config-bgp-router)# neighbor 2000:10:1::2 remote-as 62002
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 2000:10:1::2 activate
device(config-bgp-ipv4u)# end
```

This functionality is available only for IPv4 unicast address-family in the default VRF. The activation of an IPv6 peer in any other address-family, for example, VPNv4, is not supported.

Once an IPv6 neighbor is activated under IPv4 address-family, all IPv4 routes present in the BGP IPv4 unicast address-family are advertised to the IPv6 peer with the IPv6 next-hop. Similarly, IPv4 routes received from an IPv6 peer are installed in IPv4 unicast address-family with the IPv6 next-hop.

BGP EVPN in an MPLS provider network

RFC7209 specifies requirements for BGP EVPN in an MPLS provider network, highlighting the advantages of EVPN over proven L2VPN technology.

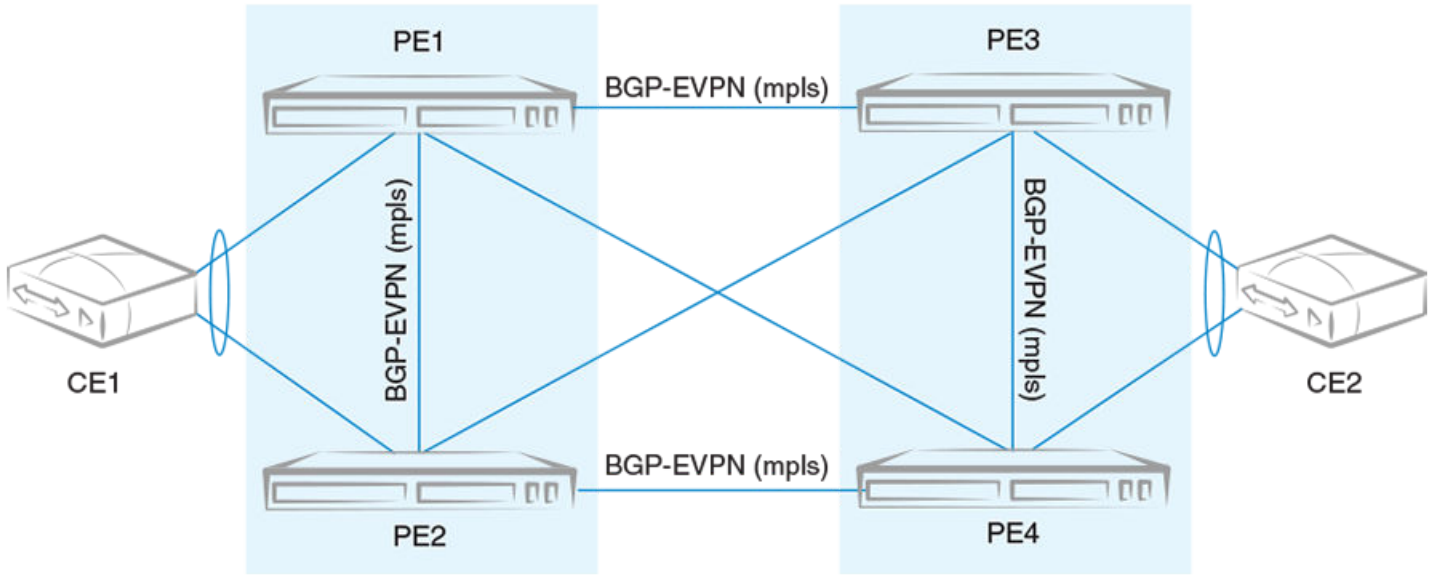
BGP EVPN in a greenfield deployment can be used as a replacement for VPLS, because of BGP EVPN's multihoming support and ability to exchange MAC addresses in the control plane. The use cases for BGP EVPN MPLS can be broadly categorized as follows:

1. Data center interconnect with control plane learning
2. Provider network deployment with forwarding plane learning (similar to VPLS)

In this release, EVPN MPLS is primarily positioned for use case (2). Layer 2 EVPN VXLAN – EVPN MPLS interconnect support does not exist. Details for supported data-center interconnect schemes are provided in BGP EVPN VXLAN data center interconnect on page 471.

The following figure shows a simplistic topology of an EVPN MPLS use case. PE1 and PE2 form an MCT cluster, and MAC addresses are learned from CE devices in the forwarding plane. These learned MAC addresses are not only exchanged between MCT peers but are also advertised to other PE devices. VLANs/BDs are extended to MPLS tunnels over inclusive-multicast routes. Note that EVPN MPLS tunnels between MCT peers and to remote PEs belong to same split-horizon groups. Consequently, neither BUM nor unicast traffic received from a remote PE is forwarded to the EVPN MPLS tunnel between MCT peers. Ingress replication is used for flooding the traffic. Each PE device in the MCT cluster receives the flooded frame from the remote PE, and the frame is flooded only to local edge ports according to the elected designated forwarder.

FIGURE 57 BGP EVPN in a provider MPLS network



Load balancing of the unicast traffic for the multihomed MAC addresses is achieved by means of aliasing, as described under "Aliasing in MPLS" in Layer 2 (MAC) route exchange on page 462.

The BGP EVPN functionalities discussed in this chapter apply also to EVPN MPLS provider networks. The following features (not limited to this set) are supported for EVPN MPLS:

- BGP/BGP-EVPN iBGP/eBGP configuration
- Static anycast gateway
- ARP suppression
- Type-1 through Type-4 EVPN routes support
- Type-5 prefix route support

NOTE
This release does not support the internetworking of VPLS with EVPN MPLS.

EVPN next-hop resolution and tunnel-EVI membership

BGP EVPN next-hop is resolved according to the encapsulation type in BGP encapsulation extended community that is attached to the route.

EVPN routes are filtered if the encapsulation type in the route differs from the encapsulation type configured for the neighbor.

The following table summarizes the next-hop address set in the EVPN MP_REACH NLRI in BGP updates originated by a router irrespective of eBGP/iBGP peering type.

TABLE 33 Next-hop address for MPLS and VXLAN

MPLS	VXLAN
BGP peering IP address	Source VTEP IP address

When there are intermediate BGP routers between the service end points, and these routers do not participate in tunnel termination and reorigination without routing in the overlay, BGP neighbors on the intermediate routers should be configured to keep the next-hop unchanged. This requirement applies to all types of tunnels, including MPLS VXLAN.

Support for IPv6 overlay next-hops is not available. The next-hop value in the EVPN routes is "IPv4 address," irrespective of IPv4 or IPv6 BGP peering.

The edge/leaf router performs next-hop resolution according to the encapsulation type. In the case of VXLAN, if VTEP discovery is enabled, VXLAN tunnels are automatically created and destroyed. For MPLS, tunnels are discovered by means of LDP/RSVP protocols. BGP resolves the EVPN MPLS next-hop with the MPLS tunnel, and routes are installed only when the next-hop is reachable. In case the MPLS tunnel becomes unreachable, the EVPN next-hop is marked as unresolved and routes are removed from the system.

The inclusive multicast route is originated for each VLAN/BD configured under EVPN, and this route triggers the extension of a corresponding VLAN/BD on the tunnel at the remote end. Import rules for inclusive multicast routes are discussed in BGP routing tables on page 449. Inclusive-multicast routes are required to carry the P-Multicast Service Interface (PMSI) tunnel attribute (RFC 6514, Sec. 5) in the path-attribute to signal the tunnel attributes. PMSI tunnel attribute fields are filled as shown in the following table.

TABLE 34 PMSI tunnel attribute fields and descriptions

Attribute Field	Description
Tunnel Type	Tunnel type is always "Ingress Replication" (value 6) for all types of tunnels. No other type of tunnel is supported currently.
MPLS label	This field carries the L2 VNI for VXLAN and the EVI label for MPLS.
Tunnel Identifier	This field carries the source VTEP IP address for VXLAN and the BGP peering IP address for MPLS.

In the case of LVTEP, both of the MCT cluster members advertise inclusive-multicast routes with different router IDs but the same next-hop. VLAN/BD membership of the tunnel is removed only when both advertising routers withdraw their inclusive-multicast routes. Therefore, consistent VLAN/BD configuration is required in an MCT cluster.

In the case of MPLS, there is always one tunnel between the MCT peers. In addition, inclusive-multicast route affects only the tunnel from which the route is originated. In order to perform load balancing of the unicast traffic, aliasing is used as discussed in the next section.

BGP EVPN in an MPLS provider network

RFC7209 specifies requirements for BGP EVPN in an MPLS provider network, highlighting the advantages of EVPN over proven L2VPN technology.

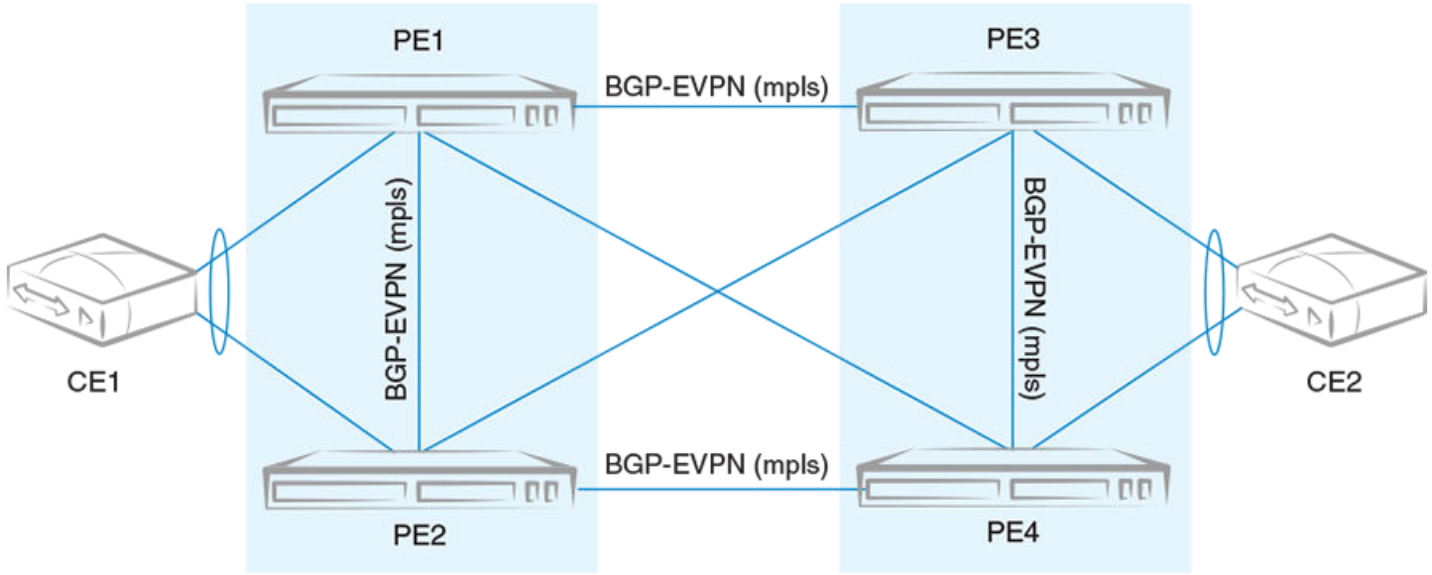
BGP EVPN in a greenfield deployment can be used as a replacement for VPLS, because of BGP EVPN's multihoming support and ability to exchange MAC addresses in the control plane. The use cases for BGP EVPN MPLS can be broadly categorized as follows:

1. Data center interconnect with control plane learning
2. Provider network deployment with forwarding plane learning (similar to VPLS)

In this release, EVPN MPLS is primarily positioned for use case (2). Layer 2 EVPN VXLAN – EVPN MPLS interconnect support does not exist. Details for supported data-center interconnect schemes are provided in BGP EVPN VXLAN data center interconnect on page 471.

The following figure shows a simplistic topology of an EVPN MPLS use case. PE1 and PE2 form an MCT cluster, and MAC addresses are learned from CE devices in the forwarding plane. These learned MAC addresses are not only exchanged between MCT peers but are also advertised to other PE devices. VLANs/BDs are extended to MPLS tunnels over inclusive-multicast routes. Note that EVPN MPLS tunnels between MCT peers and to remote PEs belong to same split-horizon groups. Consequently, neither BUM nor unicast traffic received from a remote PE is forwarded to the EVPN MPLS tunnel between MCT peers. Ingress replication is used for flooding the traffic. Each PE device in the MCT cluster receives the flooded frame from the remote PE, and the frame is flooded only to local edge ports according to the elected designated forwarder.

FIGURE 58 BGP EVPN in a provider MPLS network



Load balancing of the unicast traffic for the multihomed MAC addresses is achieved by means of aliasing, as described under "Aliasing in MPLS" in Layer 2 (MAC) route exchange on page 462.

The BGP EVPN functionalities discussed in this chapter apply also to EVPN MPLS provider networks. The following features (not limited to this set) are supported for EVPN MPLS:

- BGP/BGP-EVPN iBGP/eBGP configuration
- Static anycast gateway
- ARP suppression
- Type-1 through Type-4 EVPN routes support
- Type-5 prefix route support

NOTE
This release does not support the internetworking of VPLS with EVPN MPLS.

Layer 2 (MAC) route exchange

All dynamic MAC addresses learned on VLANs/BDs added to an EVPN configuration are exported to BGP EVPN automatically.

Routes are imported on the remote node according to the route-target match. The fields in the MAC routes are filled as shown in the following table.

TABLE 35 MAC route fields and descriptions

Field	Description
Route Distinguisher	Either the auto or manual RD value is used, depending on the VLAN/BD configuration under EVPN.
ESI	In case the MAC is learned on an MCT client interface, the ESI of the client interface is present. Otherwise this value is 0.
Ethernet Tag	This is value is 0.
MAC address	This is the static or dynamic MAC address learned locally.
IP address	For MAC route IP address, this value is absent.
MPLS Label1	This is the L2 VNI in case of VXLAN, and the EVI label for MPLS.
MPLS Label2	This is not present.

TABLE 36 MAC route fields and descriptions

Field	Description
Route Distinguisher	Either the auto or manual RD value is used, depending on the VLAN/BD configuration under EVPN.
ESI	In case the MAC is learned on an MCT client interface, the ESI of the client interface is present. Otherwise this value is 0.
Ethernet Tag	This is value is 0.
MAC address	This is the static or dynamic MAC address learned locally.
IP address	For MAC route IP address, this value is absent.

BGP MAC mobility extended community is attached to the route to carry a sticky flag and sequence number. Static MAC addresses configured on the system are advertised by BGP with a sticky flag. Routers importing a MAC route with a sticky flag install it as static and no MAC movement is allowed. In BGP best-path selection, a MAC route with a sticky flag is preferred over routes without a sticky flag, irrespective of the sequence number.

MAC address-table **show** output has been enhanced to show the remote VTEP IP address and route type as EVPN for a BGP-learned MAC over VXLAN, as in the following example.

```
device# show mac-address-table
Type Code - CL:Cluster Local MAC   CCL:Cluster Client Local MAC
            CR:Cluster Remote MAC  CCR:Cluster Client Remote MAC
VlanId/BDId   Mac-address       Type        State      Ports/LIF/PW/Tunnel
100 (V)       0000.0164.0100    Static      Inactive   Eth 0/31
101 (V)       0000.0164.0101    Static      Inactive   Eth 0/31
10 (V)        0000.0164.0010    Static      Active     Eth 0/31
10 (V)        0000.0191.0010    EVPN        Active     Tu 61441 (192.168.32.10)
10 (V)        0001.0191.0010    EVPN        Active     Tu 61441 (192.168.32.10)
10 (V)        0001.0221.0010    EVPN        Active     Tu 61441 (192.168.32.10)
```

MAC move detection and dampening

When a host moves from one port to another port on the same router, it is not considered a move. However, when the host moves from one router to another router, the MAC address undergoes a MAC move procedure. The router on which a MAC address is learned locally prefers the local address and advertises it to other routers in the network. If the MAC address is already present in BGP, the sequence number in the MAC mobility extended community is incremented in the route being advertised. A MAC route without MAC mobility extended community implicitly means sequence number 0. A router receiving a remote MAC route, with a sequence number greater than that of the locally advertised route, prefers the remote route and withdraws the local one.

MCT and LVTEP are two special cases in which a MAC move is not triggered. If the same MAC address is present in the BGP table and is learned from same next-hop or with same ESI, the MAC route is advertised with the same sequence number present in the existing route.

The frequent movement of a MAC address from one router to another router causes unnecessary churn in the network and is an indication of a malicious host or loop. When the number of moves for a given MAC address in a specified time (the default is 3 seconds) exceeds the specified number of moves (the default is 5), the MAC route is dampened. The EVPN MAC route dampening behavior differs from the BGP route flap dampening procedure specified in RFC 2439. When a MAC route is dampened, the local route is marked as best and is present in the forwarding tables. The best route selection based on sequence number is stopped until corrective action is taken. Default parameters of MAC route dampening can be modified by means of the following command under EVPN configuration mode:

```
device(config-evpn-default)# duplicate-mac-timer 5 max-count 3
```

Automatic restoration of dampened routes

According to BGP EVPN RFC 7432, once a MAC/IP route is dampened because of frequent moves, manual intervention is required to restore the route. Section 15.1 in the RFC states, "The PE MUST alert the operator and stop sending and processing any BGP MAC/IP Advertisement routes for that MAC address until a corrective action is taken by the operator."

A major drawback of this approach is that the route remains dampened and no processing of the updates is performed until the dampening state of the route is cleared manually. It may happen that after an initial frequent flap of the route, either one of the VMs goes away or the duplicate address situation is resolved. However, the route remains dampened until a network administrator intervenes. Automatic restoration of the route is desired in this case.

The following approach is taken to restore the dampened route.

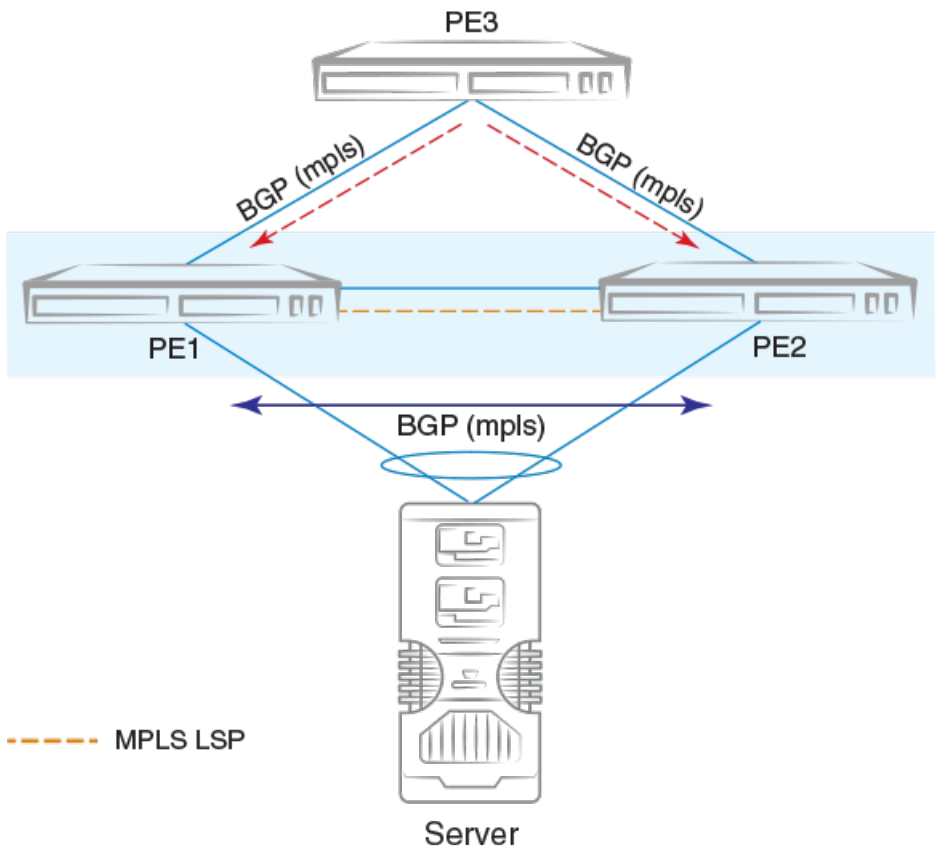
When BGP detects that only one source of the route (that is, all NLRIs are received from same next-hop) has remained and the route is dampened, the route is added to the timer list, which is processed after 5 minutes. When the timer expires, if the second source of the route is seen again, the route is removed from the timer list and remains dampened. After the timer expires, the dampening state of the route is cleared and the route is restored after best-path selection.

Aliasing in MPLS

Aliasing, as described in RFC 7432 (Sec. 8.4), is specific to MPLS. Dual-homed MAC addresses are those that are advertised with a nonzero ESI value. The membership of an ES is advertised by the ES-AD route. When a MAC address is received with a nonzero ESI, instead of the next-hop of the route being used, the number of paths and respective VLAN/BD MPLS labels are inferred from AD-per-ES and AD-per-EVI routes, respectively.

The following figure shows a typical case of aliasing in MPLS. Router R3 receives MAC routes learned on MLAG with a nonzero ESI value only from R1. However, AD-per-ES routes for the same ESI value are received from both R1 and R2. In this case, R3 may form ECMP of two paths that lead to the same MCT cluster (ES) and load-balance the traffic towards the host through both R1 and R2.

FIGURE 59 Aliasing in MPLS



Use case for AD-per-ES route

There are two use cases for AD-per-ES route on the non-MCT router:

- Faster withdrawal of MAC routes in case of ES link failures
- Load balancing of traffic for a given ES with AD-per-EVI routes

MAC/MACIP routes received with nonzero ESI value are not installed in the system unless at least one AD-per-ES route for corresponding ESI is present in BGP. Similarly, if MAC/MACIP routes are already installed and the last AD-per-ES route with corresponding ESI is withdrawn, routes are uninstalled from the forwarding plane.

Use case for AD-per-EVI route

AD-per-EVI routes are generated by MCT cluster members for each EVI configured on each ES. Because the number of AD-per-EVI routes can impose significant control-plane scaling issues, a maximum of 16 K routes is imposed.

The MPLS label field in the AD-per-EVI routes carries a per-EVI or per-ES-per-EVI MPLS label. Only per-EVI label allocation is assumed. With the assumption of a per-EVI label allocation scheme, it is not necessary to receive or originate AD-per-EVI routes for each ES or EVI. A subset of AD-per-EVI routes may suffice to identify the per-EVI label of the remote node.

BGP constructs a per-next-hop EVI-label map based on the AD-per-EVI routes. In addition, based on the reachability of AD-per-ES routes, an ECMP next-hop is constructed for Layer 2 routes.

Note that not all multihomed MAC addresses can attain forwarding plane ECMP behavior, because of limited forwarding resources. In case hardware resources are not available for ECMP in the forwarding plane, MAC addresses are sprayed across the available paths in

software according to the VLAN hashing. Therefore, it is undeterministic to know whether stream-based load balancing for a given multihomed MAC address is available in the system.

Conversational MAC learning

Conversational MAC learning is not supported.

Example output for "show mac-address-table"

The following are example outputs of the **show mac-address-table** command.

```
device# show mac-address-table
Type Code - CL:Cluster Local MAC   CCL:Cluster Client Local MAC
            CR:Cluster Remote MAC  CCR:Cluster Client Remote MAC
VlanId/BDId   Mac-address       Type            State      Ports/LIF/PW/Tunnel
100 (V)       0000.0164.0100    Static          Inactive   Eth 0/31
101 (V)       0000.0164.0101    Static          Inactive   Eth 0/31
10 (V)        0000.0164.0010    Static          Active     Eth 0/31
10 (V)        0000.0191.0010    EVPN               Active     Tu 61441 (192.168.32.10)
10 (V)        0001.0191.0010    EVPN               Active     Tu 61441 (192.168.32.10)
10 (V)        0001.0221.0010    EVPN-Static  Active     Tu 61441 (192.168.32.10)
Total MAC addresses    :  6


device# show mac-address-table evpn
Type Code - CL:Cluster Local MAC   CCL:Cluster Client Local MAC
            CR:Cluster Remote MAC  CCR:Cluster Client Remote MAC
VlanId/BDId   Mac-address       Type            State      Ports/LIF/PW/Tunnel
10 (V)        0000.0191.0010    EVPN               Active     Tu 61441 (192.168.32.10)
10 (V)        0001.0191.0010    EVPN               Active     Tu 61441 (192.168.32.10)
10 (V)        0001.0221.0010    EVPN-Static  Active     Tu 61441 (192.168.32.10)
Total MAC addresses    :  3


device# show mac-address-table count evpn
EVPN Address Count: 50


device# show mac-address-table count
Dynamic Address Count: 2
Static Address Count: 0
Internal Address Count: 0
Local Address Count    : 2
Remote Address Count   : 0
EVPN Address Count: 2
Total MAC addresses    : 6


device# show mac-address-table count interface tunnel 61441
Dynamic Address Count: 2
Static Address Count: 0
EVPN Address Count: 2
Internal Address Count: 0
Local Address Count    : 2
Remote Address Count   : 0
Total MAC addresses    : 6


device# show mac-address-table interface tunnel 61441
Type Code - CL:Cluster Local MAC   CCL:Cluster Client Local MAC
            CR:Cluster Remote MAC  CCR:Cluster Client Remote MAC
VlanId/BDId   Mac-address       Type            State      Ports/LIF/PW
10 (V)         0000.0191.0010    EVPN-Static      Active      Tu 61441 (192.168.32.10)
10 (V)         0001.0191.0010    EVPN                     Active      Tu 61441 (192.168.32.10)
Total MAC addresses    :  2
```

ARP/ND (MACIP) route exchange

ARP/ND addresses learned on the VLAN/BD added to EVPN are automatically exported into BGP. ARP/ND routes are referred to as MACIP (type 2) in RFC 7432.

The following types of ARP/ND routes are exported into BGP:

- Dynamically learned ARP/ND on VE interfaces with a local egress interface
- Statically configured ARP/ND on VE interfaces with a local egress interface
- IPv4/IPv6 addresses configured on VE interfaces
- Virtual IPv4/IPv6 addresses configured by means of VRRP/anycast gateway

If ARP/ND is learned with an egress interface as an MCT client interface, the corresponding ESI is attached to the MACIP route.

The resolution of ARP/ND received from BGP EVPN is based on the resolution of the corresponding MAC route. In addition, MAC route movement triggers the movement of MACIP routes as well. ARP/ND routes are withdrawn from BGP when the MAC resolution changes from a local egress interface to a tunnel.

Statically configured ARP/ND entries carry a sticky flag in the MAC mobility extended community and are installed as static ARP/ND on the routers importing this route. When an ARP/ND entry is sticky, the binding of the host address does not change on a local learning event.

MACIP route fields are filled as shown in the following table.

TABLE 37 MACIP route fields and descriptions

Field	Description
Route Distinguisher	Either an auto or manual RD value is used, depending on the VLAN/BD configuration under EVPN.
ESI	In case the MAC address is learned on an MCT client interface, the ESI of the client interface is present. Otherwise it is 0.
Ethernet Tag	This field is 0.
MAC address	This is the MAC address associated with the ARP/ND.
IP address	This is the IPv4/IPv6 host address.
MPLS Label1	This is the L2 VNI in case of VXLAN and the EVI label for MPLS.
MPLS Label2	This is conditionally the VRF/L3VNI, as described in the following text.

If ARP/ND is learned on a nondefault VRF and the VRF is configured to export routes into EVPN, the following behavior applies:

- The MPLS Label2 field carries the Layer 3 VNI in case of VXLAN, and the VRF label in case of MPLS. Otherwise, Label2 is not sent in the route.
- Both IP VRF RTs and VLAN/BD RTs are attached.

VRRP and anycast gateway addresses are advertised with BGP default gateway extended community; the are used for logging errors when a MACIP route is imported but the same anycast IP address is not configured.

All EVPN MACIP routes are held in the ARP suppression cache, and this database is used for ARP suppression and conversational ARP.

ARP suppression

When an ARP/ND binding for a host address is known in the control plane, the ARP suppression feature allows a router to respond to the ARP/ND requests received on edge ports according to the control plane information instead of through flooding the ARP/ND requests in the overlay network. This suppression avoids a significant amount of flooded traffic in the overlay network.

ARP/ND suppression can be enabled individually on a VLAN/BD as shown in the following examples.

```
device(config)# vlan 10
device(config-vlan-10)# suppress-arp
device(config-vlan-10)# suppress-nd

device(config)# bridge-domain 10
device(config-bridge-domain-10)# suppress-arp
device(config-bridge-domain-10)# suppress-nd
```

By default ARP/ND suppression is disabled.

> **NOTE**
> This behavior applies to both VXLAN and MPLS.

Conversational ARP learning

Conversational ARP learning is not supported.

EVPN prefix route exchange and Multi-VRF support

Extensions to RFC 7432 to support the advertisement of IP prefix routes are proposed in IETF draft "IP Prefix Advertisement in EVPN."

A new route type, Type-5, is available for exchanging IPv4 and IPv6 prefixes in the overlay network. In a multitenant Layer 3 deployment, each tenant is assigned a VRF. Prefixes in each VRF may be gateway prefixes and prefixes belonging to the tenant, and these are discovered by routing protocols such as BGP and OSPF. In order to exchange prefixes in a VRF, end-to-end routing protocol adjacency in that VRF is required between the routers hosting a given tenant. EVPN prefix route eliminates such a limitation and, similar to IP VPN, can accumulate routes across all VRFs in BGP and allow the per-VRF import of prefixes by routers depending on the route-targets.

IP VPN is MPLS-only technology, whereas EVPN prefix route exchange provides the same functionality over any type of overlay encapsulation type supported by EVPN. Prefix route exchange is supported on the VXLAN and MPLS types of BGP EVPN encapsulation and peering.

Prefix route exchange is supported on BGP EVPN encapsulation and peering.

For each tenant VRF from which prefixes are required to be exported into EVPN or imported from EVPN, the following configurations are required:

- Configure the VRF import/export route-targets for EVPN
- Select an integrated routing and bridging (IRB) interface in the VRF for routing.

The following example shows VRF RD and RT configurations in IPv4 and IPv6 address-families for the respective import and export of prefixes from and into EVPN.

```
vrf red
 rd 100:1
 evpn irb ve 10
 address-family ipv4 unicast
  route-target export 100:100 evpn
  route-target import 100:100 evpn
 !
 address-family ipv6 unicast
  route-target export 100:200 evpn
  route-target import 100:200 evpn
 !
!
```

The IRB interface is the VE interface that is used for routing after tunnel termination. The IRB interface should belong to the tenant VRF and be administratively up. It is not required to configure an IP address on the IRB interface, as shown in the following configuration example.

```
interface Ve 10
 vrf forwarding red
 no shutdown
!
```

Depending on the encapsulation, appropriate attributes for overlay are used in the EVPN prefix route advertisement, as shown in the following table.

TABLE 38 Overlay attributes and descriptions

Field	Description
Route Distinguisher	This is configured under IP VRF
ESI	This is set to 0.
IP Prefix	This is the gateway or tenant prefix in the VRF.
Gateway IP	This is set to 0.
MPLS Label	VXLAN: This is the VNI corresponding to the IRB VE interface. MPLS: This is the IP VRF MPLS label

Export route-targets are attached to prefixes exported into EVPN. Remote end routers compare configured IP VRF RTs against RTs in the route and, in case of a match, prefix routes are imported into EVPN and then into IP VRF tables. In case RTs in a prefix route match multiple VRFs, routes from the EVPN table are imported into each matching VRF.

After the import into IP VRF, the next-hop resolution of the prefix route is not performed in the VRF. Instead, overlay next-hop is used and the prefix route is programmed with the exit interface as a tunnel.

In the case of VXLAN, the advertising router attaches the MAC address of the IRB interface as part of the MAC extended community of the BGP router in the prefix route. This MAC address is used as the next-hop MAC by the remote routers. In the forwarding plane, after tunnel decapsulation, routing on the inner frame is performed, because the outer DA MAC is the IRB interface MAC.

In the case of MPLS, MAC address of the router is not attached, because the inner VRF label in the frame causes routing in the corresponding VRF.

Symmetric or asymmetric routing

Depending on how VLANs/BDs are extended on the leaf nodes, symmetric or asymmetric routing may be observed. In the case of symmetric routing, routing on a VLAN/BD occurs only at a single router/leaf in the network. On the other hand, when the same VLAN/BD is extended on multiple routers, in the presence of distributed anycast gateway, each leaf node performs routing for traffic entering from local edge ports, causing routing to occur asymmetrically, depending on the direction of traffic.

Import/export route-map filtering

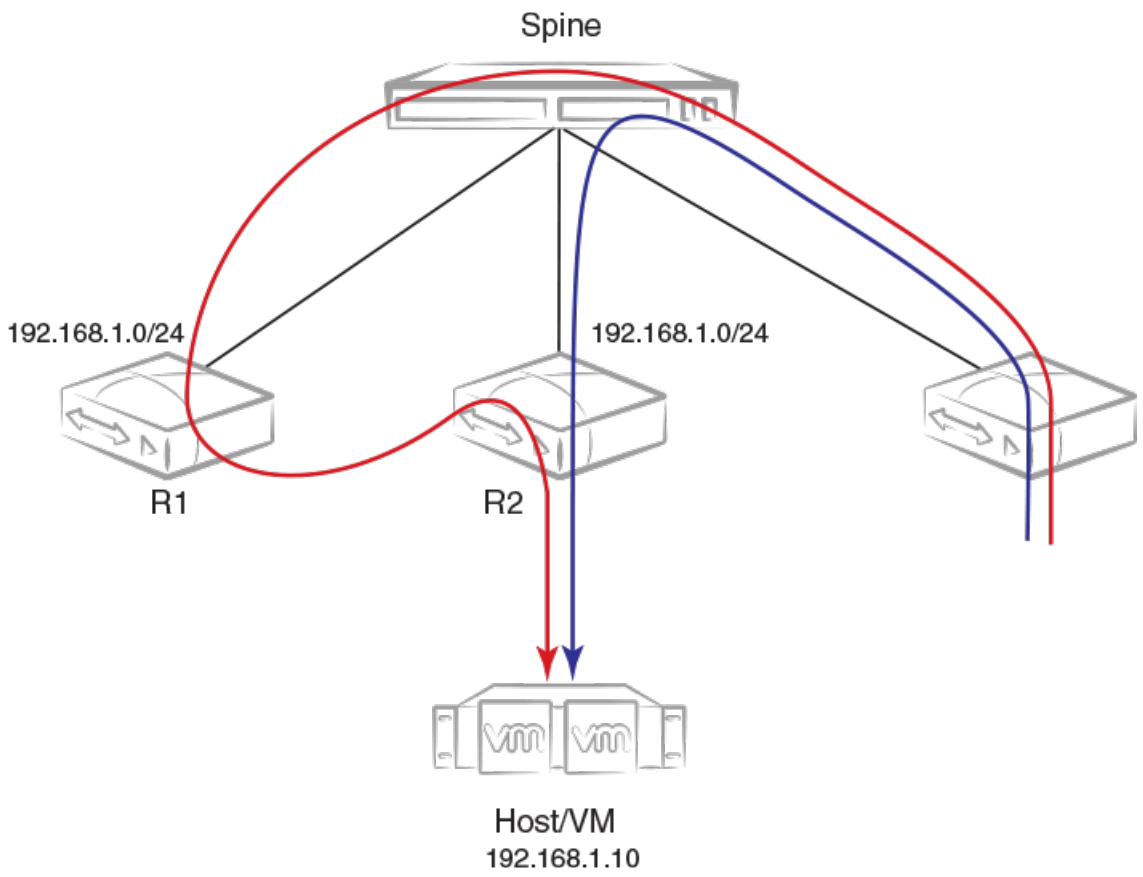
The route-map-based filtering of routes imported into IPVRF or exported into EVPN is supported. An import or export route-map can be applied under IPv4 or IPv6 address-family under a VRF configuration, as shown in the following example configuration.

```
device(vrf-red-ipv4-unicast)# import map my-import-filter evpn
device(vrf-red-ipv4-unicast)# export map my-export-filter evpn
```

Conversion of MACIP to host route to avoid traffic tromboning

In a typical IP Fabric network where distributed anycast gateway is configured on multiple leaf nodes, the same connected prefix is advertised by multiple nodes. A leaf node that does not extend the same VLAN/BD, but has extended the VRF, imports these prefixes, forms ECMP, and load balances the traffic across the tunnels. Because of ECMP, traffic may take a suboptimal path as shown in the following figure.

FIGURE 60 Avoiding suboptimal paths by converting MACIP routes to host routes



In order to avoid the suboptimal forwarding, a router extending the tenant VRF accepts MACIP routes matching the IP VRF RT in the EVPN table and converts those MACIP routes to IPv4 /32 prefix routes or IPv6 /128 or routes and imports them into the IP VRF table.

Anycast/VRRP IPv4 and IPv6 prefix routes are advertised with BGP default gateway extended community.

In the current release, because conversational ARP/ND is not supported, all converted IPv4 /32 host routes are installed in the forwarding plane. On the other hand, even though prefixes are received with default-gateway extended community, they are installed in the forwarding plane as normal prefix routes pointing to the ECMP next-hop.

By default, the conversion of MACIP routes to host routes is enabled. However, it can be disabled by means of the following configuration under BGP EVPN address-family.

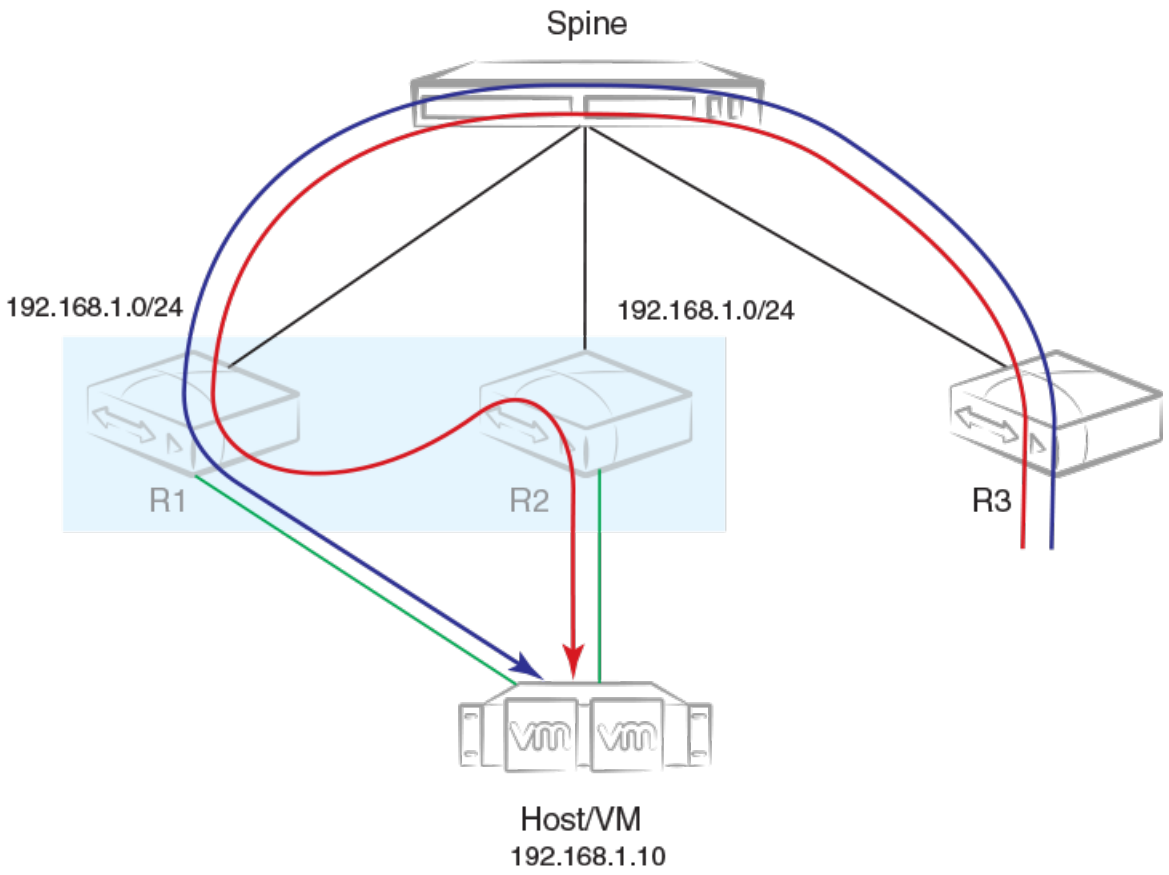
```
router bgp
  address-family l2vpn evpn
    suppress ipv4 host-route
    !
!
```

Avoiding traffic tromboning in an MCT cluster

In an EVPN VXLAN network, Type-5 prefix routes are advertised by an MCT cluster with a source VTEP IP address that is common within the cluster. Also, each route carries a router-MAC address that is system-MAC unique to an MCT node. The remote node installing the prefix routes advertised by the MCT peer chooses one of the paths in the forwarding plane. However, an intermediate router may further load balance the traffic depending on the destination LVTEP IP address.

In the following figure, traffic towards Host/VM is load balanced by R3 across R1–R2 ECMP paths (the same next-hop but with different router-MAC addresses). However, the same traffic gets load balanced again by Spine across R1 and R2. Because of the underlay load balancing, Layer 3 traffic with the router-MAC of R2 may land on R1 and conversely. Because the destination MAC after VXLAN tunnel termination belongs to the MCT peer, traffic is Layer 2-switched to the MCT peer and is then routed there. This causes traffic tromboning.

FIGURE 61 Avoiding traffic tromboning for EVPN Layer 3 traffic in an MCT cluster



This issue is resolved by exchanging the router-MAC address within the MCT cluster and installing the router MAC address of the MCT peer for routing based on the following example configuration.

```
vrf red
  evpn irb ve 10 cluster-gateway
```

When a peer gateway configuration is present for an EVPN IRB VE interface for the VRF, Layer 3 traffic destined for the MCT peer's router MAC in the corresponding VLAN/BD is routed locally.

BGP EVPN VXLAN data center interconnect

This section details the BGP EVPN VXLAN support for data center interconnect (DCI).

Data center Layer 2 interconnect between EVPN VXLAN and EVPN MPLS or VPLS is not supported. However, EVPN VXLAN and VPLS can coexist on the router, although a given VLAN/BD should not be extended into EVPN and be configured for VPLS.

The following sections address a variety of DCI interconnect scenarios.

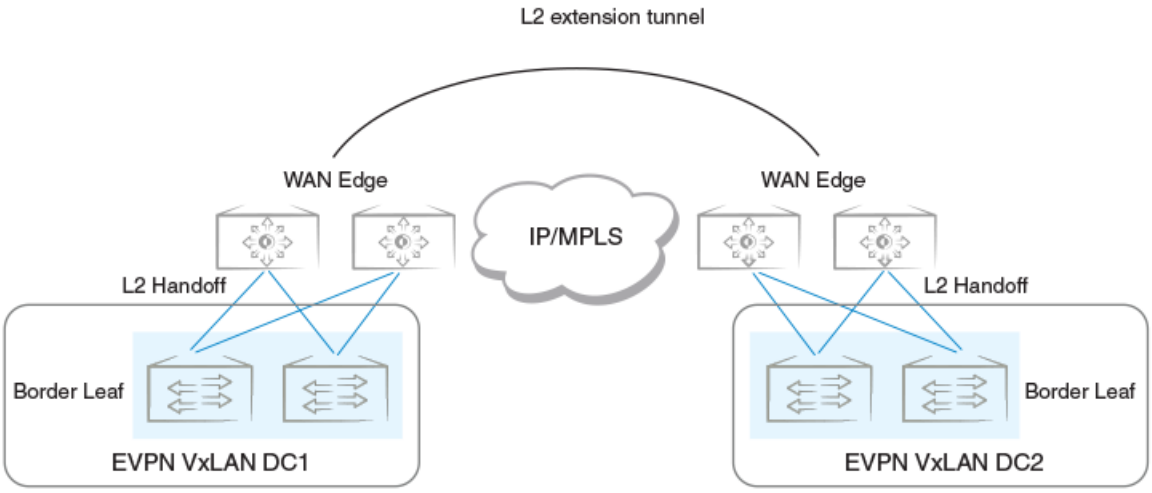
Layer 2 and Layer 3 control-plane extension

In this scenario, VXLAN tunnels are extended by means of a border leaf instead of getting terminated. This extension is primarily the spine functionality being provided by the border leaf, except that the control and forwarding planes are extended over the WAN/core in the case of a border leaf.

Layer 2 handoff

In this scenario, VXLAN tunnels are terminated at the border leaf, as shown in the following figure. Depending on the interconnect technology being used between data centers extending over the WAN (such as VPLS, VXLAN Layer 2 extension, or any other interconnect scheme), the border leaf bridges the two domains through forwarding plane learning without any control plane extension.

FIGURE 62 Layer 2 handoff-based DCI

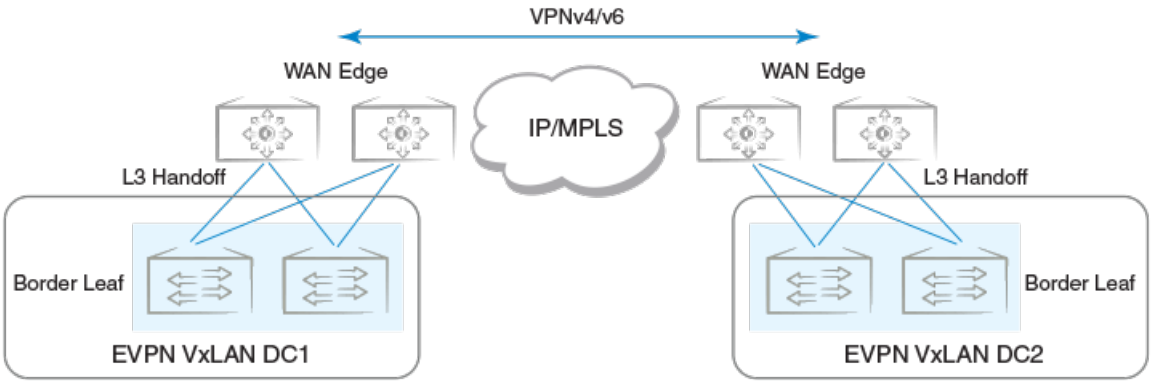


In this scenario, VXLAN tunnels are terminated at the border leaf. Depending on the interconnect technology being used between data centers extending over the WAN, the border leaf bridges the two domains through forwarding plane learning without any control plane extension.

Layer 3 handoff

In the case of Layer 3, all VRFs in the data center are terminated on the border leaf and traffic is routed towards the WAN, as shown in the following figure. Similarly, Layer 3 traffic received from the WAN is routed and forwarded over tunnels in the data center. Because Layer 3 routes across multiple VRFs can be imported into a single (interconnecting) VRF, all possible VRFs in a given data center do not have to be configured on the border leaf.

FIGURE 63 Layer 3 handoff-based DCI

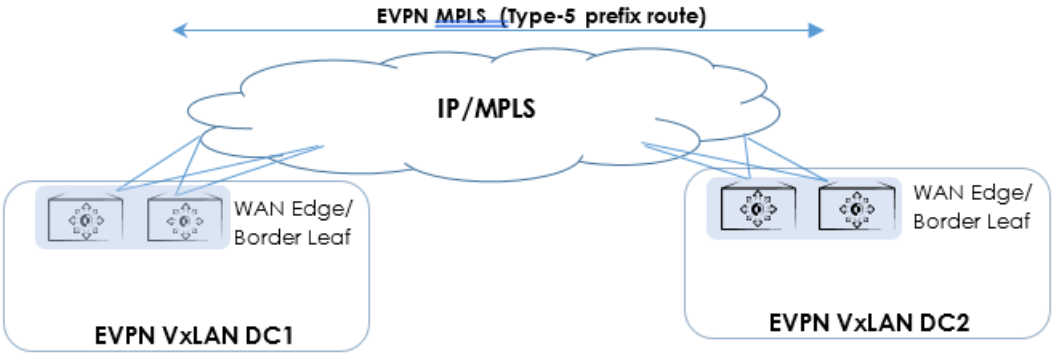


In the case of Layer 3, all VRFs in the data center are terminated on the border leaf and traffic is routed towards the WAN. Similarly, Layer 3 traffic received from the WAN is routed and forwarded over tunnels in the data center. Because Layer 3 routes across multiple VRFs can be imported into a single (interconnecting) VRF, all possible VRFs in a given data center do not have to be configured on the border leaf.

EVPN Layer 3 interconnect

In this scenario, EVPN VXLAN tunnels within the data center are terminated on the WAN edge, as shown in the following figure. In the BGP EVPN control plane, only Type-5 prefix routes are imported from EVPN VXLAN peers and are reoriginated towards EVPN MPLS peers. Similarly, in the other direction, EVPN Type-5 prefix routes are imported from EVPN MPLS peers and are reoriginated towards EVPN VXLAN peers within the data center.

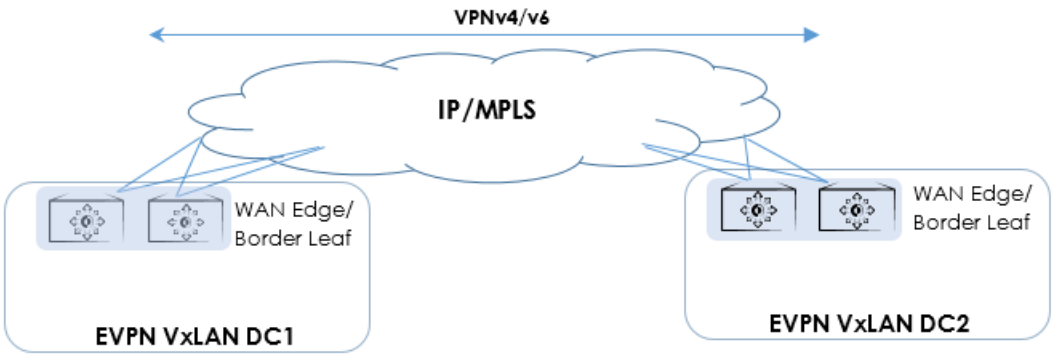
FIGURE 64 EVPN Type-5 prefix route termination and reorigination on the WAN edge



EVPN VPNv4/v6 interconnect

In this scenario, EVPN Type-5 prefix routes from the VXLAN data center are terminated on the WAN-edge device, and are redistributed into VPNv4/v6 address-family, as shown in the following figure. Similarly, IPv4/v6 prefix routes imported into VRFs from VPNv4/v6 are exported into EVPN and are advertised to EVPN VXLAN peers by means of Type-5 prefix routes.

FIGURE 65 Data center Layer 3 interconnect with EVPN VXLAN and VPNv4/v6



Static anycast gateway

Static anycast gateway functionality provides support for seamless VM mobility across the leaf switches in IP Fabric deployments.

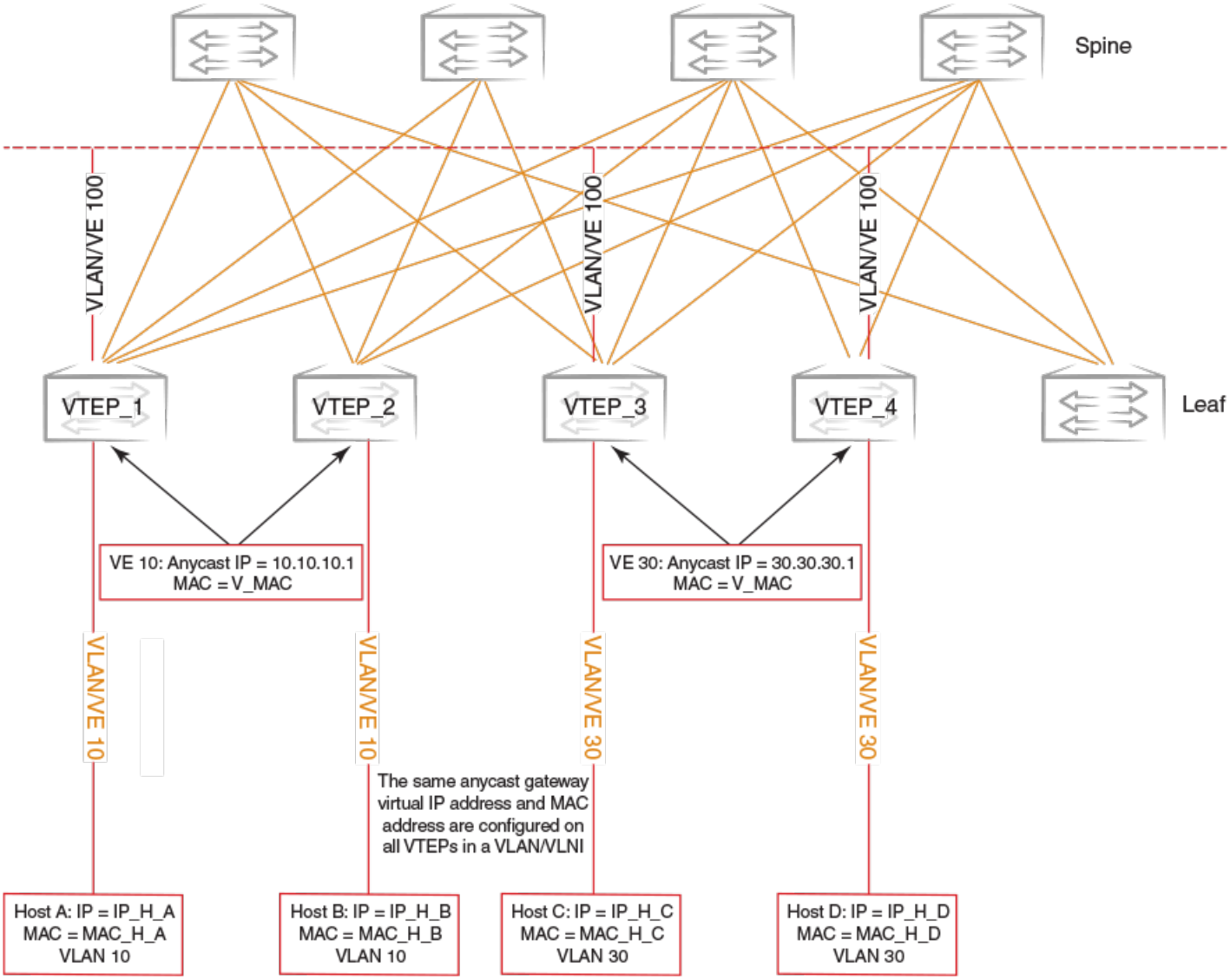
Hosts attached to the leaf switches are configured with a default gateway, which is typically the IP address of the leaf switch in the VLAN facing the host. If this host moves to another leaf switch, the host has to be reconfigured with the IP address of the new leaf switch to which it is now connected as its default gateway.

To make this movement of host from one leaf switch to another seamless, all the leaf switches are configured with the same anycast IP address and the associated virtual MAC (VMAC or anycast MAC) address for a given VLAN/BD. This configuration allows any leaf switch to behave as the default gateway for the host and allows for the most optimal forwarding behavior.

The ingress leaf switch recognizes the VMAC address as its own MAC address and does Layer 3 forwarding.

As shown in the following figure, for VLAN 10, all leaf switches are configured with the same MAC address and IP address.

FIGURE 66 Static anycast gateway



When the host sends an ARP request for the gateway IP address on VLAN 10, the ingress leaf switch intercepts the ARP request and responds with the VMAC address associated with anycast IP. This behavior is controlled for each VE interface.

> **NOTE**
> Static anycast gateway is recommended only in the presence of a BGP EVPN control plane.

In order to configure static anycast gateway, configure the VMAC address first by specifying the default MAC address or an arbitrary unicast MAC address as in the following IPv4 and IPv6 examples. The default MAC address values are 02e0.5200.0100 and 02e0.5200.0200 for IPv4 and IPv6, respectively.

```
device(config)# ip anycast-gateway-mac default-mac
device(config)# ipv6 anycast-gateway-mac default-mac

device(config)# ip anycast-gateway-mac 0000.0101.0101
device(config)# ipv6 anycast-gateway-mac 0000.0101.0102
```

The anycast gateway IP address can be configured under the VE interface for VLANs/BDs, as in the following example.

```
device(config)# vlan 100
device(config-vlan-100)# router-interface ve 100
device(config-vlan-100)# int ve 100
device(config-if-Ve-100)# ip anycast-address 100.0.0.1/24
device(config-if-Ve-100)# ipv6 anycast-address 1000::1/24
```

NOTE
An IPv4 and IPv6 anycast MAC address cannot be configured as the same MAC address.

The configured anycast gateway address can be seen in the following examples.

```
device# show ip anycast-gateway
Gateway mac: 02e0.5200.0100
Ve10          1.1.1.0/24                                  Inactive (Interface Down)
Ve100         100.0.0.1/24                                Active

device# show ipv6 anycast-gateway
Gateway mac: 02e0.5200.0200
Ve100         1000::1/24                                  Active
```

ATTENTION
It is recommended that ARP suppression be configured on VLANs/BDs if static anycast gateway is configured on the corresponding VE interface. Otherwise, duplicate ARP responses to the gateway IP address are observed.

IP unnumbered interface

In a typical Layer 3 Clos network, routers are directly connected and require the assignment of IP addresses to each pair of routers, typically from the same subnet.

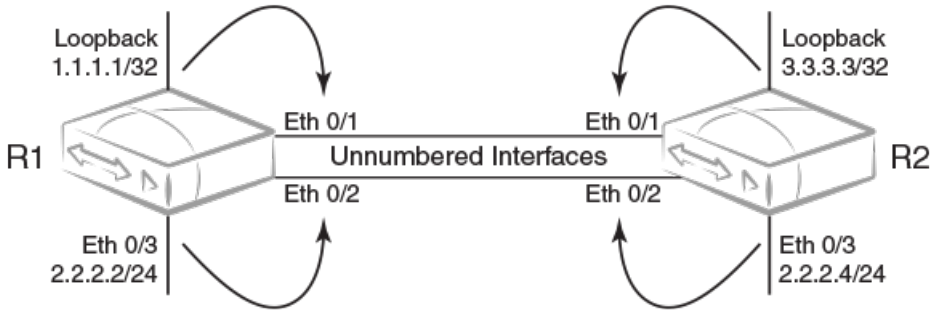
With large numbers of routers and redundant Layer 3 interfaces, a significant number of IP addresses are consumed just to configure the network itself. Using /31 masks reduces the consumption of addresses, but two IP addresses are still consumed per interface. Using unnumbered interfaces greatly reduces the number of IP addresses consumed in the configuration of the network.

This feature borrows an IP address from another Layer 3 interface already configured on the router. This address is used as a source address in the Layer 3 packets that are sent out the unnumbered interface. The interface from which the IP address is borrowed is called the donor interface. The following points highlight some of the key aspects of the use and functionality of this feature:

- Only physical interfaces can be configured as unnumbered interfaces. Unnumbered interfaces are not supported for virtual Ethenet (VE) or switched virtual interface (SVI) interfaces.
- The donor interface can be any other Layer 3 interface (Ethernet/VE/loopback).
- Unnumbered interface support is provided for IPv4 address family. Consequently, only an IPv4 address can be borrowed for an unnumbered interface.
- Once the interface is configured as an unnumbered interface, it is treated as a point-to-point interface and there can be only one remote neighbor attached to the interface.
- Because no network subnet is associated with the unnumbered interface, ARP is not supported on an unnumbered interface.
- Unnumbered interfaces are not supported in the management VRF, because routing protocols (OSPF/BGP) are not enabled in the management VRF.

The following diagram illustrates the overall concept of unnumbered interfaces and donor interfaces.

FIGURE 67 IP unnumbered and donor interfaces



- Interfaces Ethernet 0/1 and Ethernet 0/2 on R1 and R2 are unnumbered interfaces.
- Interfaces Ethernet 0/1 on R1 and R2 borrow IPv4 addresses from a loopback interface.
- Interfaces Ethernet 0/2 on R1 and R2 borrow IPv4 addresses from physical interface Ethernet 0/3.
- The end points of the unnumbered interface may or may not be in the same subnet.

The following functionalities are required to support Layer 3 over unnumbered interfaces:

- Neighbor discovery
- Host routes to reach neighbors over unnumbered interfaces
- Routes using unnumbered interfaces as next-hops

NOTE
For a simple three-stage IP Fabric, IP unnumbered interfaces can be used. For a five-stage IP Fabric, numbered interfaces are highly recommended, although IP unnumbered interfaces can be used in five-stage IP Fabric deployments if third-party devices are not included in the design. Extreme does not recommend using a mix of unnumbered and numbered interfaces within an IP Fabric.

High availability

BGP EVPN graceful restart (GR) capabilities are supported as follows:

- BGP EVPN GR helper is supported.
- BGP EVPN GR router restart is not supported.
- BGP Process-restart is not supported in BGP EVPN.

OSPFv2

- OSPFv2 overview 478
- Autonomous System 478
- OSPFv2 components and roles 479
- Reduction of equivalent AS external LSAs 481
- Algorithm for AS external LSA reduction 483
- Enabling OSPFv2 483
- Backbone area 483
- Assigning OSPFv2 areas 485
- Area range 485
- Area types 486
- Stub area and totally stubby area 487
- Not-so-stubby area (NSSA) 488
- Assigning interfaces to an area 493
- Assigning interfaces to an area 493
- Link state advertisements 494
- Link state advertisements 494
- Configuring an MD5 password and authentication change hold time for an OSPFv2 interface 495
- Virtual links 495
- Default route origination 498
- External route summarization 499
- SPF timers 499
- Modifying Shortest Path First timers 500
- OSPFv2 administrative distance 501
- OSPFv2 LSA refreshes 502
- Support for OSPF RFC 2328 Appendix E 503
- OSPFv2 graceful restart 504
- OSPFv2 stub router advertisement 507
- OSPFv2 Shortest Path First throttling 508
- IETF RFC and internet draft support 508
- OSPFv2 non-stop routing 508
- Synchronization of critical OSPFv2 elements 510
- Standby module operations 512
- OSPFv2 distribute list 512
- OSPFv2 route redistribution 514
- Redistributing routes into OSPFv2 516
- Load sharing 516
- OSPFv2 type 3 LSA filtering 518
- Interface types to which the reference bandwidth does not apply 522
- Changing the reference bandwidth for the cost on OSPFv2 interfaces 522
- OSPFv2 over VRF 523
- OSPF VRF-Lite for customer edge routers 524
- OSPFv2 in a VCS environment 524
- OSPFv2 considerations and limitations 525
- Configuring the OSPFv2 Max-Metric Router LSA 526
- Re-enabling OSPFv2 compatibility with RFC 1583 527
- Enabling OSPFv2 in a VCS environment 527
- Changing default settings 529
- Disabling and re-enabling OSPFv2 event logging 529

- Understanding the effects of disabling OSPFv2 530
- Displaying OSPFv2 results 531
- Displaying OSPFv2 results 535

OSPFv2 overview

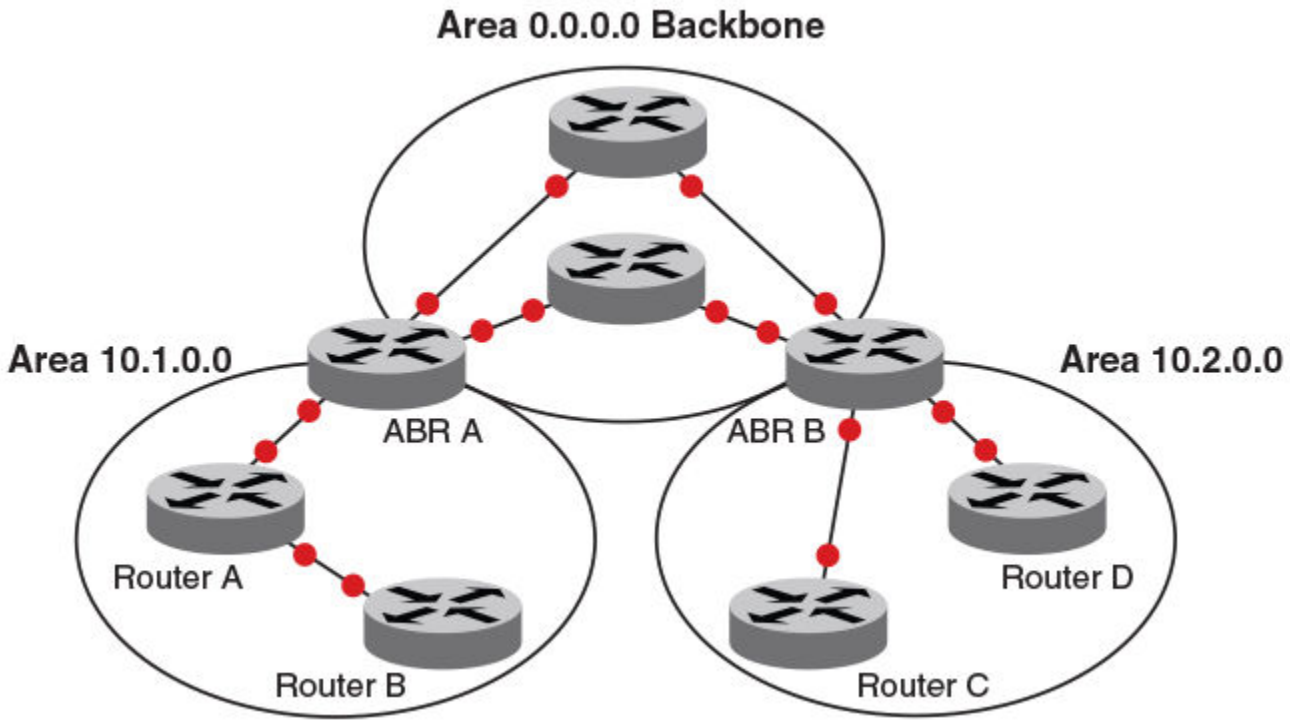
Open Shortest Path First Version 2 (OSPFv2) is a link-state routing protocol that uses link-state advertisements (LSAs) to update neighboring routers about a router's interfaces. Each router maintains an identical area-topology database to determine the shortest path to any neighboring router.

OSPF is built upon a hierarchy of network components and areas. The highest level of the hierarchy is the autonomous system. An autonomous system is defined as a number of networks, all of which share the same routing and administration characteristics. A backbone area forms the core of the network, connecting all other areas. Details of these and other OSPF components are provided below.

Autonomous System

An Autonomous System can be divided into multiple areas. Each area represents a collection of contiguous networks and hosts. Areas limit the amount of advertisements sent within the network. This is known as flooding. An area is represented in OSPFv2 by either an IP address or a number.

FIGURE 68 OSPF operating in a network



NOTE
For details of components and virtual links, refer to OSPFv2 components and roles on page 479 and Virtual links on page 495, respectively.

Once OSPFv2 is enabled on the system, the user assigns an IP address or number as the *area ID* for each area. The area ID is representative of all IP addresses (subnets) on a router port. Each port on a router can support one area.

OSPFv2 components and roles

OSPFv2 can be configured on either a point-to-point or broadcast network.

Devices can take a variety of roles in an OSPFv2 topology, as discussed below.

Area Border Routers

An OSPF router can be a member of multiple areas. Routers with membership in multiple areas are known as Area Border Routers (ABRs). All ABRs must have either a direct or indirect link to an OSPF backbone area (also known as area 0 or area 0.0.0.0). Each ABR maintains a separate topological database for each area the router is in. Each topological database contains all LSA databases for each router within a given area. The routers within the same area have identical topological databases. An ABR is responsible for forwarding routing information or changes among its border areas.

For more information on OSPFv2 areas, refer to the *OSPFv2 areas* section.

Autonomous System Boundary Routers

An Autonomous System Boundary Router (ASBR) is a router that is running multiple protocols and serves as a gateway to routers outside the OSPF domain and those operating with different protocols. The ASBR is able to import and translate different protocol routes into OSPF through a process known as redistribution.

Designated routers

In an OSPF broadcast network, OSPF elects one router to serve as the designated router (DR) and another router on the segment to act as the backup designated router (BDR). This minimizes the amount of repetitive information that is forwarded on the network. OSPF forwards all messages to the designated router.

On broadcast networks such as LAN links, all routers on the LAN other than the DR and BDR form full adjacencies with the DR and BDR and pass LSAs only to them. The DR forwards updates received from one neighbor on the LAN to all other neighbors on that same LAN. One of the main functions of a DR is to ensure that all the routers on the same LAN have identical LSDBs. Therefore, on broadcast networks, an LSDB is synchronized between a DROther (a router that is not a DR or a BDR) and its DR and BDR.

NOTE
In an OSPF point-to-point network, where a direct Layer 3 connection exists between a single pair of OSPF routers, there is no need for designated or backup designated routers.

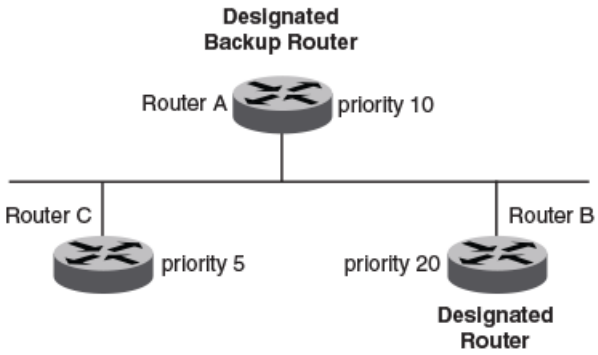
Without the need for Designated and Backup Designated routers, a point-to-point network establishes adjacency and converges faster. The neighboring routers become adjacent whenever they can communicate directly. In contrast, in broadcast and non-broadcast multi-access (NBMA) networks, the Designated Router and Backup Designated Router become adjacent to all other routers attached to the network.

In a network with no designated router and no backup designated router, the neighboring router with the highest priority is elected as the DR, and the router with the next highest priority is elected as the BDR, as shown in the figure below. Priority is a configurable option at the interface level; refer to the **ip ospf priority** command in the *Extreme NetIron Command Reference*.

In a network with no designated router and no backup designated router, the neighboring router with the highest priority is elected as the DR, and the router with the next highest priority is elected as the BDR, as shown in the figure below. Priority is a configurable option at the interface level; refer to the **ip ospf priority** command in the *Extreme Network OS Command Reference*.

In a network with no designated router and no backup designated router, the neighboring router with the highest priority is elected as the DR, and the router with the next highest priority is elected as the BDR, as shown in the figure below. Priority is a configurable option at the interface level; refer to the **ip ospf priority** command in the *Command Reference*.

FIGURE 69 Designated and backup router election



If the DR goes off line, the BDR automatically becomes the DR. The router with the next highest priority becomes the new BDR.

If two neighbors share the same priority, the router with the highest router ID is designated as the DR. The router with the next highest router ID is designated as the BDR. The DR and BDRs are recalculated after the OSPF protocol is disabled and re-enabled by means of the **[no] router ospf** command.

> NOTE
> By default, the device's router ID is the IP address configured on the lowest numbered loopback interface. If the device does not have a loopback interface, the default router ID is the lowest numbered IP address configured on the device.

When multiple routers on the same network are declaring themselves DRs, then both the priority and router ID are used to select the designated router and backup designated routers.

The DR and BDR election process is performed when one of the following events occurs:

- An interface is in a waiting state and the wait time expires.
- An interface is in a waiting state and receives a hello packet that addresses the BDR.
- A change in the neighbor state occurs, such as the following:
  - A neighbor state transitions from ATTEMPT state to a higher state.
  - Communication to a neighbor is lost.
  - A neighbor declares itself to be the DR or BDR for the first time.

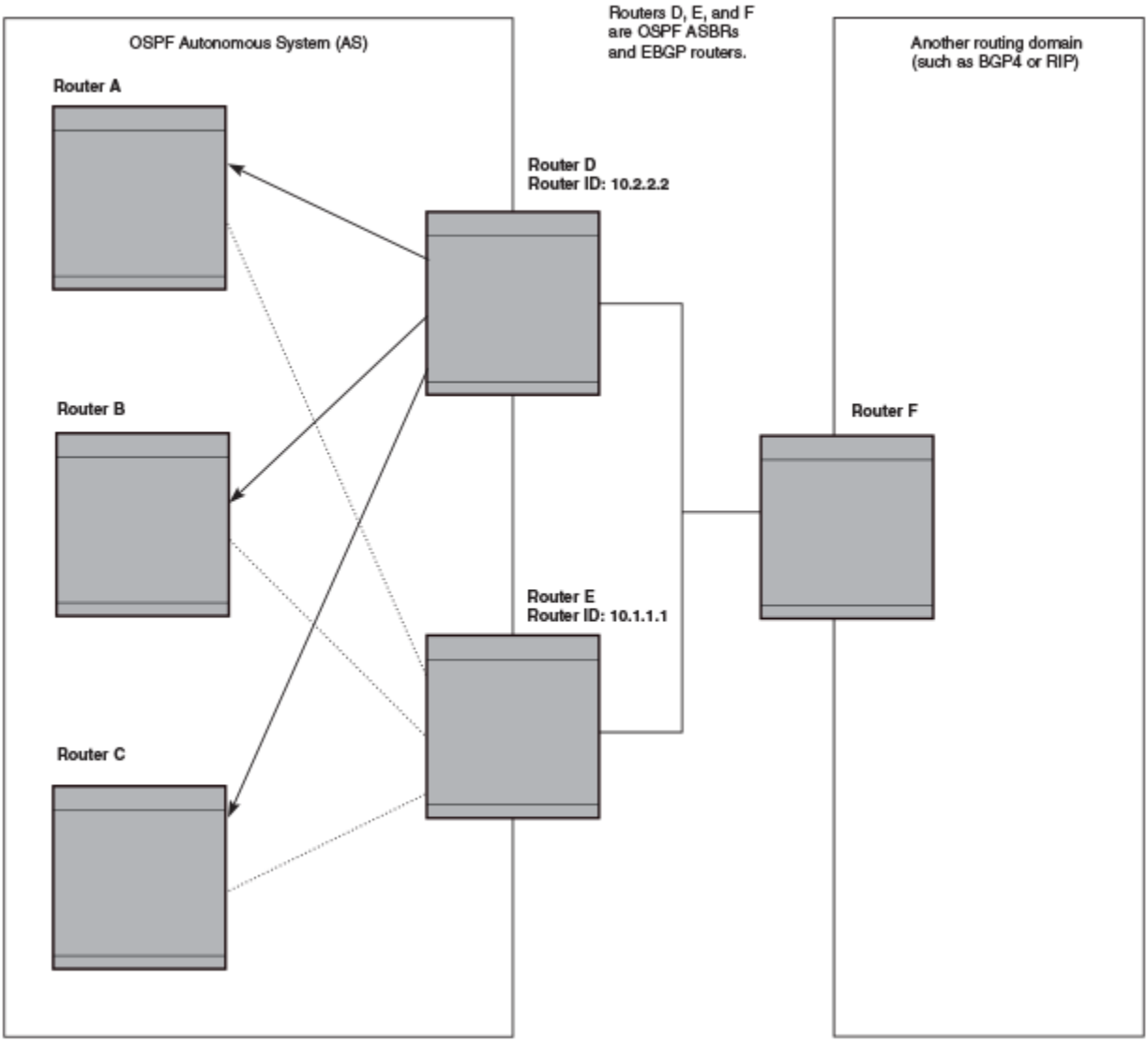
Reduction of equivalent AS external LSAs

An OSPF ASBR uses AS External link advertisements (AS External LSAs) to originate advertisements of a route learned from another routing domain, such as a BGP4 or RIP domain. The ASBR advertises the route to the external domain by flooding AS External LSAs to all the other OSPF devices (except those inside stub networks) within the local OSPF Autonomous System (AS).

In some cases, multiple ASBRs in an AS can originate equivalent LSAs. The LSAs are equivalent when they have the same cost, the same next hop, and the same destination. The device optimizes OSPF by eliminating duplicate AS External LSAs in this case. The device with the lower router ID flushes the duplicate External LSAs from its database and thus does not flood the duplicate External LSAs into the OSPF AS. AS External LSA reduction, therefore, reduces the size of the link state database on the device. The AS External LSA reduction is described in RFC 2328

In this example, Routers D and E are OSPF ASBRs, and thus communicate route information between the OSPF AS, which contains Routers A, B, and C, and another routing domain, which contains Router F. The other routing domain is running another routing protocol, such as BGP4 or RIP. Routers D, E, and F, therefore, are each running both OSPF and either BGP4 or RIP.

FIGURE 70 AS external LSA reduction



Notice that both Router D and Router E have a route to the other routing domain through Router F.

OSPF eliminates the duplicate AS External LSAs. When two or more devices are configured as ASBRs have equal-cost routes to the same next-hop router in an external routing domain, the ASBR with the highest router ID floods the AS External LSAs for the external domain into the OSPF AS, while the other ASBRs flush the equivalent AS External LSAs from their databases. As a result, the overall volume of route advertisement traffic within the AS is reduced and the devices that flush the duplicate AS External LSAs have more memory for other OSPF data. Because Router D has a higher router ID than Router E, Router D floods the AS External LSAs for Router F to Routers A, B, and C. Router E flushes the equivalent AS External LSAs from its database.

Algorithm for AS external LSA reduction

The AS external LSA reduction example shows the normal AS External LSA reduction feature. The behavior changes under the following conditions:

- There is one ASBR advertising (originating) a route to the external destination, but one of the following happens:
  - A second ASBR comes on-line
  - A second ASBR that is already on-line begins advertising an equivalent route to the same destination.

In either case above, the router with the higher router ID floods the AS External LSAs and the other router flushes its equivalent AS External LSAs. For example, if Router D is offline, Router E is the only source for a route to the external routing domain. When Router D comes on-line, it takes over flooding of the AS External LSAs to Router F, while Router E flushes its equivalent AS External LSAs to Router F.

- One of the ASBRs starts advertising a route that is no longer equivalent to the route the other ASBR is advertising. In this case, the ASBRs each flood AS External LSAs. Since the LSAs either no longer have the same cost or no longer have the same next-hop router, the LSAs are no longer equivalent, and the LSA reduction feature no longer applies.
- The ASBR with the higher router ID becomes unavailable or is reconfigured so that it is no longer an ASBR. In this case, the other ASBR floods the AS External LSAs. For example, if Router D goes off-line, then Router E starts flooding the AS with AS External LSAs for the route to Router F.

Enabling OSPFv2

A number of steps are required when enabling OSPFv2 on a device.

Consider the following when enabling OSPFv2 on a device.

- Redistribution must be enabled on devices configured to operate as ASBRs.
- All device ports must be assigned to one of the defined areas on an OSPF device. When a port is assigned to an area, all corresponding subnets on that port are automatically included in the assignment.

1. Enter the **router ospf** command in global configuration mode to enable OSPF on the device.
2. Enter the **router ospf** command in RBridge ID configuration mode to enable OSPF on the device.
3. Assign the areas to which the device will be attached.
4. Assign individual interfaces to the OSPF areas.
5. Assign a virtual link to any ABR that does not have a direct link to the OSPF backbone area.
6. Refer to Changing default settings on page 529.

Backbone area

The backbone area (also known as area 0 or area 0.0.0.0) forms the core of OSPF networks. All other areas should be connected to the backbone area either by a direct link or by virtual link configuration. Routers that have interfaces in both backbone area and (at least one) non-backbone area are called Area Border Routers (ABR). Inter area routing happens via ABRs.

The backbone area is the logical and physical structure for the OSPF domain and is attached to all non-zero areas in the OSPF domain.

The backbone area is responsible for distributing routing information between non-backbone areas. The backbone must be contiguous, but it does not need to be physically contiguous; backbone connectivity can be established and maintained through the configuration of virtual links.

Setting up the backbone area

The following example sets up the backbone area.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 10
   ```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPF on the device.

   ```
   device(onfig-rbridge-id-10)# router ospf
   ```

4. Enter the **area** command and specify *0.0.0.0* to configure the backbone area. .

   ```
   device(config-router-ospf-vrf-default-vrf)# area 0.0.0.0
   ```

5. Enter the **exit** command until you return to global configuration mode.

   ```
   device(config-router-ospf-vrf-default-vrf)# exit
   ```

6. Enter the **interface vlan** command followed by the VLAN number to create a VLAN.

   ```
   device(config)# interface vlan 1001
   ```

7. Enter the **rbridge-id** command followed by the RBridge ID to enter RBridge configuration mode.

   ```
   device(config)# rbridge-id 10
   ```

8. Enter the **interface ve** command followed by the VLAN number to enter interface configuration mode.

   ```
   device(config-rbridge-id-10)# interface Ve 1001
   ```

9. Enter the **ip address** operand followed by the IP address/subnet for the interface.

   ```
   device(config-Ve-1001 # ip address 101.1.1.1/24
   ```

10. Enter the **ip ospf area** command followed by the area ID to assign the interface to this area.

    ```
    device(config-Ve-1001)# ip ospf area 0.0.0.0
    ```

```
device# configure terminal
device(config)# rbridge 10
device(config)# ip router-id 10.11.12.13
device(config-rbridge-id-10)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 0.0.0.0
device(config-router-ospf-vrf-default-vrf)# exit
device(config-rbridge-id-10)# exit
device(config)# interface vlan 1001
device(config-Vlan-1001)# rbridge 10
device(config-rbridge-id-10)# interface Ve 1001
device(config-Ve-1001 # ip address 101.1.1.1/24
device(config-Ve-1001)# ip ospf area 0.0.0.0
```

Assigning OSPFv2 areas

Areas can be assigned as OSPFv2 areas.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config)# router ospf
   ```

3. Enter the **area** command to define an OSPFv2 area ID.

   ```
   device(config-ospf-router)# area 0
   device(config-router-ospf-vrf-default-vrf)# area 0
   ```

4. Enter the **area** command to define a second OSPFv2 area ID.

   ```
   device(config-ospf-router)# area 10.1.1.1
   device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1
   ```

The following example assigns an OSPFv2 ID to two areas. One of the areas is assigned by decimal number. The second area is assigned by IP address.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# area 0
device(config-ospf-router)# area 10.1.1.1

device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 0
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1
```

Area range

You can further consolidate routes at an area boundary by defining an area range. The area range allows you to assign an aggregate address to a range of IP and IPv6 addresses.

This aggregate value becomes the address that is advertised instead of all the individual addresses it represents being advertised. Only this aggregate or summary address is advertised into other areas instead of all the individual addresses that fall in the configured range. Area range configuration can considerably reduce the number of Type 3 summary LSAs advertised by a device. You have the option of adding the cost to the summarized route. If you do not specify a value, the cost value is the default range metric calculation for the generated summary LSA cost. You can temporarily pause route summarization from the area by suppressing the type 3 LSA so that the component networks remain hidden from other networks.

You can assign up to 32 ranges in an OSPF area.

You can assign up to 32 ranges in an OSPFv2 area.

Assigning an area range

Ranges for an area can be assigned. Ranges allow a specific IP address and mask to represent a range of IP addresses within an area, so that only that reference range address is advertised to the network, instead of all the addresses within that range. Each area can have up to 32 range addresses.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf

device(config-rbridge-id-122)# router ospf
```

4. Enter the **area range** command, specifying an area ID, and enter the range. Repeat as necessary.

```
device(config-ospf-router)# area 10.0.0.10 range 10.45.0.0 10.255.0.0
device(config-ospf-router)# area 10.0.0.20 range 10.45.0.0 10.255.0.0

device(config-router-ospf-vrf-default-vrf)# area 10.0.0.10 range 10.45.0.0 255.255.0.0
device(config-router-ospf-vrf-default-vrf)# area 10.0.0.20 range 10.45.0.0 255.255.0.0
```

5. Enter the **area range** command, specifying an area ID, and enter the range. Use the **cost** parameter to specify the cost value for the area range. Repeat as necessary.

```
device(config-ospf-router)# area 10.0.0.10 range 10.45.0.0 10.255.0.0 cost 20
device(config-ospf-router)# area 10.0.0.20 range 10.45.0.0 10.255.0.0 cost 20
```

The following example defines an area range for subnets on 10.0.0.10 and 10.0.0.20.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# area 10.0.0.10 range 10.45.0.0 10.255.0.0
device(config-ospf-router)# area 10.0.0.20 range 10.45.0.0 10.255.0.0

device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 10.0.0.10 range 10.45.0.0 10.255.0.0
device(config-router-ospf-vrf-default-vrf)# area 10.0.0.20 range 10.45.0.0 10.255.0.0
```

The following example defines an area range for subnets on 10.0.0.10 and 10.0.0.20 and sets a cost of 20 for the area range.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# area 10.0.0.10 range 10.45.0.0 10.255.0.0 cost 20
device(config-ospf-router)# area 10.0.0.20 range 10.45.0.0 10.255.0.0 cost 20
```

Area types

OSPFv2 areas can be normal, a stub area, a totally stubby area (TSA), or a not-so-stubby area (NSSA).

- Normal: OSPFv2 devices within a normal area can send and receive external link-state advertisements (LSAs).

- Stub: OSPFv2 devices within a stub area cannot send or receive external LSAs. In addition, OSPFv2 devices in a stub area must use a default route to the area's Area Border Router (ABR) to send traffic out of the area.
- NSSA: The Autonomous System Boundary Router (ASBR) of an NSSA can import external route information into the area.
  - ASBRs redistribute (import) external routes into the NSSA as type 7 LSAs. Type 7 External LSAs are a special type of LSA generated only by ASBRs within an NSSA, and are flooded to all the routers within only that NSSA.
  - ABRs translate type 7 LSAs into type 5 External LSAs, which can then be flooded throughout the autonomous system. The NSSA translator converts a type 7 LSA to a type 5 LSA if F-bit and P-bit are set and there is a reachable forwarding address. You can configure summary-addresses on the ABR of an NSSA so that the ABR converts multiple type 7 external LSAs received from the NSSA into a single type 5 external LSA.

  When an NSSA contains more than one ABR, OSPFv2 elects one of the ABRs to perform the LSA translation for NSSA. OSPFv2 elects the ABR with the highest router ID. If the elected ABR becomes unavailable, OSPFv2 automatically elects the ABR with the next highest router ID to take over translation of LSAs for the NSSA. The election process for NSSA ABRs is automatic.
- TSA: Similar to a stub area, a TSA does not allow summary routes in addition to not having external routes.

Stub area and totally stubby area

A stub area is an area in which advertisements of external routes are not allowed, reducing the size of the database. A totally stubby area (TSA) is a stub area in which summary link-state advertisement (type 3 LSAs) are not sent. A default summary LSA, with a prefix of 0.0.0.0/0 is originated into the stub area by an ABR, so that devices in the area can forward all traffic for which a specific route is not known, via ABR.

A stub area disables advertisements of external routes. By default, the ABR sends summary LSAs (type 3 LSAs) into stub areas. You can further reduce the number of LSAs sent into a stub area by configuring the device to stop sending type 3 LSAs into the area. You can disable the summary LSAs to create a TSA when you are configuring the stub area or after you have configured the area.

The ABR of a totally stubby area disables origination of summary LSAs into this area, but still accepts summary LSAs from OSPF neighbors and floods them to other neighbors.

When you enter the **area stub** command with the **no-summary** keyword and specify an area to disable the summary LSAs, the change takes effect immediately. If you apply the option to a previously configured area, the device flushes all the summary LSAs it has generated (as an ABR) from the area with the exception of the default summary LSA originated. This default LSA is needed for the internal routers, since external routes are not propagated to them.

> NOTE
> Stub areas and TSAs apply only when the device is configured as an Area Border Router (ABR) for the area. To completely prevent summary LSAs from being sent to the area, disable the summary LSAs on each OSPF router that is an ABR for the area.

Disabling summary LSAs for a stub area

LSAs can be disabled for a stub area.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **area stub** command, specifying an area and a cost, followed by the **no-summary** parameter to set an additional cost on a specified stub area and prevent any Type 3 and Type 4 summary LSAs from being injected into the area.

```
device(config-ospf-router)# area 40 stub 99 no-summary
```

```
device(config-router-ospf-vrf-default-vrf)# area 40 stub 99 no-summary
```

The following example configures a stub area, specifying a cost of 99 and preventing any Type 3 and Type 4 summary LSAs from being injected into the area.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# area 40 stub 99 no-summary

device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 40 stub 99 no-summary

device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 40 stub 99 no-summary
```

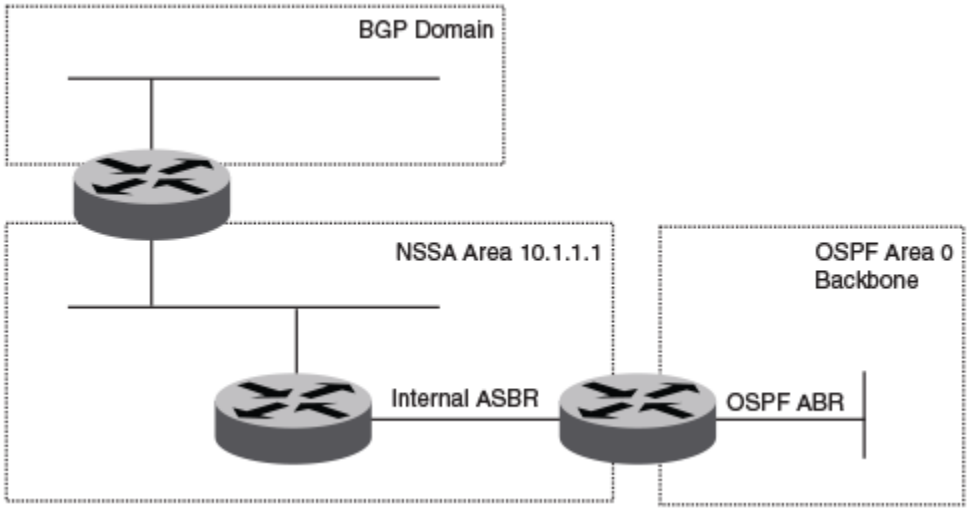
Not-so-stubby area (NSSA)

The OSPFv2 not-so-stubby area (NSSA) enables you to configure OSPFv2 areas that provide the benefits of stub areas, but that also are capable of importing external route information. OSPFv2 does not flood external routes from other areas into an NSSA, but does translate and flood route information from the NSSA into other areas such as the backbone. Since external routes are not published, a Type 7 default LSA with a prefix of ::/0 and a cost of 10 is originated into the NSSA area by the ABR to ensure that traffic passes through.

NSSAs are especially useful when you want to summarize type 5 External LSAs (external routes) before forwarding them into an OSPFv2 area. The OSPFv2 specification prohibits summarization of type 5 LSAs and requires OSPFv2 to flood type 5 LSAs throughout a routing domain. When you configure an NSSA, you can specify a summary-address for aggregating the external routes that the NSSA's ABR exports into other areas.

The figure below shows an example of an OSPFv2 network containing an NSSA.

FIGURE 71 OSPF network containing an NSSA



This example shows two routing domains, a BGP domain and an OSPF domain. The ASBR inside the NSSA imports external routes from BGP into the NSSA as type 7 LSAs, which the ASBR floods throughout the NSSA.

The ABR translates the type 7 LSAs into type 5 LSAs. If a summary-address is configured for the NSSA, the ABR also summarizes the LSAs into an aggregate LSA before flooding the type 5 LSAs into the backbone.

Because the NSSA is partially stubby the ABR does not flood external LSAs from the backbone into the NSSA. To provide access to the rest of the Autonomous System (AS), the ABR generates a default type 7 LSA into the NSSA.

ABRs of an NSSA area can be configured with the no-summary parameter to prevent the generation of type 3 and type 4 summary LSAs into the area. The only exception is the default type 3 LSA, with a prefix of 0.0.0.0/0. The default type 7 LSA is not originated in this case.

Appendix-E implementation for OSPFv2 Type7 LSA

The Link State ID in an NSSA Type 7 LSA is usually set to the described network's IP address. This leads to a limitation that we cannot redistribute multiple networks having the same address, but different masks, since they will result in same Link State ID. Appendix-E clause in RFC 2328 allows a way to overcome this limitation by setting network's host bits in the Link State ID.

When there is a need to originate more than one Type 7 OSPF NSSA LSA with same network address, LSA for the least specific prefix (least prefix length) uses Link State ID same as that of the network address. OSPF NSSA LSAs with more specific prefixes (higher prefix lengths) use Link State ID with their host-bits set.

This feature is always enabled.

The following example displays a static route with LSA for the least specific prefix (least prefix length) for OSPF NSSA LSA configuration.

```
device# show ip ospf database link-state nssa

Area ID         Type LS ID          Adv Rtr         Seq(Hex)        Age  Cksum
1               NSSA 10.0.0.0       47.3.200.1      0x80000001      146  0x2335
  LSA Header:  age: 146, options: 0x0 , seq-nbr: 0x80000001, length: 36
  Network Mask: 255.0.0.0
  TOS 0:  metric_type: 2, metric: 10
          forwarding_address: 100.1.2.3
          external_route_tag: 0
```

```
Area ID         Type LS ID           Adv Rtr         Seq(Hex)        Age  Cksum
1               NSSA 10.0.255.255    47.3.200.1      0x80000001      123  0x2335
  LSA Header:  age: 123, options: 0x0 , seq-nbr: 0x80000001, length: 36
  Network Mask: 255.255.0.0
  TOS 0:  metric_type: 2, metric: 10
          forwarding_address: 100.1.2.3
          external_route_tag: 0
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# ip route 10.0.0.0/16 47.3.1.200
```

Configuring an NSSA

OSPFv2 areas can be defined as NSSA areas with modifiable parameters.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config)# router ospf
   ```

   ```
   device(config-rbridge-id-122)# router ospf
   ```

4. Enter the **area nssa** command and specify an area address and a cost.

   ```
   device(config-ospf-router)# area 10.1.1.1 nssa 1
   ```

   ```
   device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 nssa 1
   ```

   Area 10.1.1.1 is defined as an NSSA.

The following example configures OSPF area 10.1.1.1 as an NSSA.

```
device# configure terminal
device(router ospf
device(config-ospf-router)# area 10.1.1.1 nssa 1

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 nssa 1

device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 nssa 1
```

Configuring a summary-address for the NSSA

If you want the ABR that connects the NSSA to other areas to summarize the routes in the NSSA before translating them into type 5 LSAs and flooding them into the other areas, configure an address range summary-address. The ABR creates an aggregate value based on the address range. The aggregate value becomes the address that the ABR advertises instead of advertising the individual addresses represented by the aggregate. You can configure up to 32 ranges in an OSPFv2 area.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
device(config-rbridge-id-122)# router ospf
```

4. Enter the **area nssa** command, specifying an area and a cost.

```
device(config-ospf-router)# area 10.1.1.1 nssa 10
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 nssa 10
```

5. Enter the **summary-address** command, followed by the IP address and mask for the summary route.

```
device(config-ospf-router)# summary-address 10.10.1.0    10.10.2.0
device(config-router-ospf-vrf-default-vrf)# summary-address 10.10.1.0    10.10.2.0
```

The following example configures a summary-address in NSSA 10.1.1.1.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# area 10.1.1.1 nssa 10
device(config-ospf-router)# summary-address 10.10.1.0    10.10.2.0

device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 nssa 10
device(config-router-ospf-vrf-default-vrf)# summary-address 10.10.1.0    10.10.2.0

device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 nssa 10
device(config-router-ospf-vrf-default-vrf)# summary-address 10.10.1.0 10.10.2.0
```

Configuring the translator role for a NSSA

The translator role by default is candidate. If you want the ABR to unconditionally assume the role of a NSSA translator, configure the **translator-always** option.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **area nssa** command, specifying an area. Use the **translator-always** parameter to configure the translator role, causing the router to unconditionally assume the role of a NSSA translator.

```
device(config-router-ospf-vrf-default-vrf)# area 1.1.1.1 nssa translator-always
```

The following example configures a summary-address in NSSA 1.1.1.1 and configures the router to unconditionally assume the role of a NSSA translator.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 1.1.1.1 nssa translator-always
```

Configuring the OSPF default metric

The OSPF default metric and the metric type of default route originated into NSSA area can be specified.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config-rbridge-id-122)# router ospf
   ```

4. Enter the **area nssa** command, specifying an area. Use the **default-information-metric** keyword with the **metric-type type-1** keywords to specify that the metric of a neighbor is the cost between itself and the router plus the cost of using this router for routing to the rest of the world.

   ```
   device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 default-information-metric metric-type
   type1
   ```

The following example specifies that the metric type of the default route originated into NSSA area 10.1.1.1 is type 1.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 default-information-metric metric-type type1
```

Preventing an NSSA ABR from generating external LSAs into a NSSA area

If you want an ASBR to generate type-5 LSA into normal areas and to prevent it from generating type-7 LSA into the NSSA area, you can configure the no redistribution option.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config-rbridge-id-122)# router ospf
   ```

4. Enter the **area nssa** command, specifying an area and a cost. Use the **no-redistribution** parameter to prevent an NSSA ABR from generating external LSAs into a NSSA area.

   ```
   device(config-router-ospf-vrf-default-vrf)# area 6 nssa 4 no-redistribution
   ```

The following example prevents an NSSA ABR from generating external LSAs into a NSSA area.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 6 nssa 4 no-redistribution
```

Assigning interfaces to an area

Once you define OSPFv2 areas, you can assign interfaces to the areas. All device ports must be assigned to one of the defined areas on an OSPFv2 device. When a port is assigned to an area, all corresponding subnets on that port are automatically included in the assignment.

To assign a loopback interface to an area with the IP address of 10.5.0.0, perform the following task:

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface** command and specify an interface.

   ```
   device(config)# interface loopback 2
   ```

3. Enter the **ip ospf area** command followed by the IP address of the area.

   ```
   device(config-lbif-2)# ip ospf area 10.5.0.0
   ```

   ```
   device(config-Loopback-2)# ip ospf area 10.5.0.0
   ```

   If you want to set an interface to passive mode, use the **ip ospf passive** command. If you want to block flooding of outbound LSAs on specific OSPF interfaces, use the **ip ospf database-filter all out** command.(Refer to the *Extreme NetIron Command Reference* for details.)

The following example assigns a loopback interface to an area with the IP address of 10.5.0.0.

```
device# configure terminal
device(config)# interface loopback 2
device(config-lbif-2)# ip ospf area 10.5.0.0

device# configure terminal
device(config)# interface loopback 2
device(config-Loopback-2)# ip ospf area 10.5.0.0
```

Assigning interfaces to an area

Once you define OSPFv2 areas, you can assign interfaces to the areas. All device ports must be assigned to one of the defined areas on an OSPF device. When a port is assigned to an area, all corresponding subnets on that port are automatically included in the assignment.

To assign a tengigabitethernet interface 101/0/1 to an area with the IP address of 10.5.0.0, perform the following task:

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 101
   ```

3. Enter the **interface** command and specify an interface.

```
device(config-rbridge-id-101)# interface tengigabitethernet 101/0/1
```

4. Issue the **ip ospf area** command followed by the IP address of the area.

```
device(conf-if-te-101/0/1)# ip ospf area 10.5.0.0
```

   If you want to set an interface to passive mode, use the **ip ospf passive** command. If you want to block flooding of outbound LSAs on specific OSPFv2 interfaces, use the **ip ospf database-filter all out** command.(Refer to the *Network OS Command Reference* for details.)

The following example assigns a tengigabitethernet interface to an area with the IP address of 10.5.0.0.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# interface tengigabitethernet 101/0/1
device(conf-if-te-101/0/1)# ip ospf area 10.5.0.0
```

Link state advertisements

Extreme devices support the following types of LSAs, which are described in RFC 2328 and 3101:

- Router link
- Network link
- Summary link
- Autonomous system summary link
- AS external link
- Not-So-Stubby Area (NSSA) external link
- Grace LSAs

Communication among areas is provided by means of link state advertisements (LSAs). The LSAs supported for each area type are as follows:

- Backbone (area 0) supports LSAs 1, 2, 3, 4, 5, and 7.
- Nonbackbone area supports LSAs 1, 2, 3, 4, and 5.
- Stub area supports LSAs 1, 2, and 3.
- Totally stubby area (TSA) supports LSAs 1 and 2, and also supports a single LSA 3 per ABR, advertising a default route.
- No so stubby area (NSSA) supports LSAs 1, 2, 3, and 7.

Link state advertisements

Communication among areas is provided by means of link state advertisements (LSAs). The LSAs supported for each area type are as follows:

- Backbone (area 0) supports LSAs 1, 2, 3, 4, 5, and 7.
- Backbone (area 0) supports LSAs 1, 2, 3, 4, and 5.
- Nonbackbone, supports LSAs 1, 2, 3, 4, and 5.
- Stub area supports LSAs 1, 2, and 3.
- Totally stubby area (TSA) supports LSAs 1 and 2, and also supports a single LSA 3 per ABR, advertising a default route.

- No so stubby area (NSSA) supports LSAs 1, 2, 3, and 7.

Configuring an MD5 password and authentication change hold time for an OSPFv2 interface

An MD5 password and authentication change hold time can be configured. This task sets an MD5 password and sets an authentication change hold time for an OSPFv2 Ethernet interface

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface** command and specify an interface.

```
device(config)# interface ethernet 1/1
device(config)# interface ethernet 0/1
```

3. Enter the **ip ospf md5-authentication** command with the **key-activation-wait-time** parameter, and specify a time interval, to set the time that OSPFv2 waits before activating a new MD5 key.

```
device(conf-if-eth-1/1)# ip ospf md5-authentication key-activation-wait-time 240
device(conf-if-eth-0/1)# ip ospf md5-authentication key-activation-wait-time 240
```

4. Enter the **ip ospf md5-authentication** command with the **key-id** and **key** parameters to set the MD5 key ID and a password.

```
device(conf-if-eth-1/1)# ip ospf md5-authentication key-id 22 key myospfpassword
device(conf-if-eth-0/1)# ip ospf md5-authentication key-id 22 key myospfpassword
```

The following example sets the time that OSPFv2 waits before activating a new MD5 key to 240 seconds on an Ethernet interface, and sets the MD5 key ID to 22 and a password “myospfpassword” on an Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/1
device(conf-if-eth-1/1)# ip ospf md5-authentication key-activation-wait-time 240
device(conf-if-eth-1/1)# ip ospf md5-authentication key-id 22 key myospfpassword

device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# ip ospf md5-authentication key-activation-wait-time 240
device(conf-if-eth-0/1)# ip ospf md5-authentication key-id 22 key myospfpassword
```

Virtual links

All ABRs must have either a direct or indirect link to the OSPFv2 backbone area (0.0.0.0 or 0). If an ABR does not have a physical link to the area backbone, the ABR can configure a virtual link to another router within the same area, which has a physical connection to the area backbone.

The path for a virtual link is through an area shared by the neighbor ABR (router with a physical backbone connection), and the ABR requires a logical connection to the backbone.

Two parameters fields must be defined for all virtual links—transit area ID and neighbor router:

- The transit area ID represents the shared area of the two ABRs and serves as the connection point between the two routers. This number should match the area ID value.

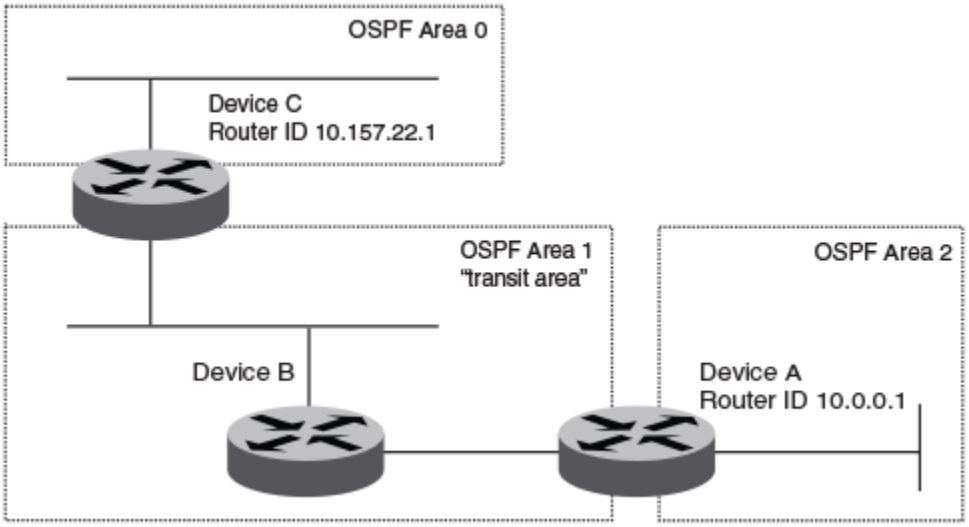
- The neighbor router field is the router ID (IP address) of the router that is physically connected to the backbone, when assigned from the router interface requiring a logical connection. When assigning the parameters from the router with the physical connection, be aware that the router ID is the IP address of the router requiring a logical connection to the backbone.

  NOTE
  By default, a device's router ID is the IP address configured on the lowest numbered loopback interface. If the device does not have a loopback interface, the default router ID is the lowest numbered IP address configured on the device. When you establish an area virtual link, you must configure it on both of the routers (both ends of the virtual link).

Virtual links cannot be configured in stub areas and NSSAs.

The following figure shows an OSPF area border router, Device A, that is cut off from the backbone area (area 0). To provide backbone access to Device A, you can add a virtual link between Device A and Device C using Area 1 as a transit area. To configure the virtual link, you define the link on the router that is at each end of the link. No configuration for the virtual link is required on the routers in the transit area.

FIGURE 72 Defining OSPF virtual links within a network



Configuring virtual links

If an Area Border Router (ABR) does not have a physical link to a backbone area, a virtual link can be configured between that ABR and another device within the same area that has a physical link to a backbone area.

A virtual link is configured, and a virtual link endpoint on two devices, ABR1 and ABR2, is defined.

1. On ABR1, enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
device(config-rbridge-id-122)# router ospf
```

4. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-ospf-router)# area 0
device(config-router-ospf-vrf-default-vrf)# area 0
```

5. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-ospf-router)# area 1
device(config-router-ospf-vrf-default-vrf)# area 1
```

6. Enter the **area virtual-link** command and the ID of the OSPFv2 device at the remote end of the virtual link to configure the virtual link endpoint.

```
device(config-ospf-router)# area 1 virtual-link 10.2.2.2
device(config-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.2.2.2
```

7. On ABR2, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

8. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 104
```

9. Enter the **router ospf** command to enter OSPFv2 router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
device(config-rbridge-id-104)# router ospf
```

10. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-ospf-router)# area 1
device(config-router-ospf-vrf-default-vrf)# area 1
```

11. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-ospf-router)# area 2
device(config-router-ospf-vrf-default-vrf)# area 2
```

12. Enter the **area virtual-link** command and the ID of the OSPFv2 device at the remote end of the virtual link to configure the virtual link endpoint.

```
device(config-ospf-router)# area 1 virtual-link 10.1.1.1
device(config-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1
```

The following example configures a virtual link between two devices.

```
ABR1:
device1# configure terminal
device1(config)# router ospf
device1(config-ospf-router)# area 0
device1(config-ospf-router)# area 1
device1(config-ospf-router)# area 1 virtual-link 10.2.2.2

ABR2:
device2# configure terminal
device2(config)# router ospf
device2(config-ospf-router)# area 1
device2(config-ospf-router)# area 2
device2(config-ospf-router)# area 1 virtual-link 10.1.1.1

ABR1:
device1# configure terminal
device1(config)# rbridge-id 122
device1(config-rbridge-id-122)# router ospf
device1(config-router-ospf-vrf-default-vrf)# area 0
device1(config-router-ospf-vrf-default-vrf)# area 1
device1(config-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.2.2.2

ABR2:
device2# configure terminal
device2(config)# rbridge-id 104
device2(config-rbridge-id-104)# router ospf
device2(config-router-ospf-vrf-default-vrf)# area 1
device2(config-router-ospf-vrf-default-vrf)# area 2
device2(config-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1

ABR1:
device1# configure terminal
device1(config)# router ospf
device1(config-router-ospf-vrf-default-vrf)# area 0
device1(config-router-ospf-vrf-default-vrf)# area 1
device1(config-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.2.2.2

ABR2:
device2# configure terminal
device2(config)# router ospf
device2(config-router-ospf-vrf-default-vrf)# area 1
device2(config-router-ospf-vrf-default-vrf)# area 2
device2(config-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1
```

Default route origination

When the device is an OSPFv2 Autonomous System Boundary Router (ASBR), you can configure it to automatically generate a default external route into an OSPFv2 routing domain.

By default, a device does not advertise the default route into the OSPFv2 domain. If you want the device to advertise the OSPFv2 default route, you must explicitly enable default route origination. When you enable OSPFv2 default route origination, the device advertises a type 5 default route that is flooded throughout the autonomous system, with the exception of stub areas.

The device advertises the default route into OSPFv2 even if OSPFv2 route redistribution is not enabled, and even if the default route is learned through an iIBGP neighbor when default-information-originate is configured. The device does not, however, originate the default route if the active default route is learned from an OSPFv2 device in the same domain.

> **NOTE**
> The device does not advertise the OSPFv2 default route, regardless of other configuration parameters, unless you explicitly enable default route origination.

If default route origination is enabled and you disable it, the default route originated by the device is flushed. Default routes generated by other OSPFv2 devices are not affected. If you re-enable the default route origination, the change takes effect immediately and you do not need to reload the software.

External route summarization

An ASBR can be configured to advertise one external route as an aggregate for all redistributed routes that are covered by a specified address range.

When you configure a summary address range, the range takes effect immediately. All the imported routes are summarized according to the configured summary address range. Imported routes that have already been advertised and that fall within the range are flushed out of the autonomous system and a single route corresponding to the range is advertised.

If a route that falls within a configured summary address range is imported by the device, no action is taken if the device has already advertised the aggregate route; otherwise, the device advertises the aggregate route. If an imported route that falls within a configured summary address range is removed by the device, no action is taken if there are other imported routes that fall within the same summary address range; otherwise, the aggregate route is flushed.

You can configure up to 32 summary address ranges. The device sets the forwarding address of the aggregate route to 0 and sets the tag to 0. If you delete a summary address range, the advertised aggregate route is flushed and all imported routes that fall within the range are advertised individually. If an external link-state database (LSDB) overflow condition occurs, all aggregate routes and other external routes are flushed out of the autonomous system. When the device exits the external LSDB overflow condition, all the imported routes are summarized according to the configured summary address ranges.

> NOTE
> If you use redistribution filters in addition to summary address ranges, the device applies the redistribution filters to routes first, and then applies them to the summary address ranges.
>
> NOTE
> If you disable redistribution, all the aggregate routes are flushed, along with other imported routes.
>
> NOTE
> This option affects only imported, type 5 external LSA routes. A single type 5 LSA is generated and flooded throughout the autonomous system for multiple external routes. Type 7-route redistribution is not affected by this feature. All type 7 routes will be imported (if redistribution is enabled). To summarize type 7 LSAs or exported routes, use NSSA address range summarization.
>
> NOTE
> Only imported, type 5 external LSA routes are affected. A single type 5 LSA is generated and flooded throughout the autonomous system for multiple external routes.

SPF timers

The device uses an SPF delay timer and an SPF hold-time timer to calculate the shortest path for OSPFv2 routes. The values for both timers can be changed.

- SPF delay: When the device receives a topology change, it waits before starting a Shortest Path First (SPF) calculation. By default, the device waits zero seconds. You can configure the SPF delay to a value from 0 through 65535 seconds. If you set the SPF delay to 0 seconds, the device immediately begins the SPF calculation after receiving a topology change.

- SPF hold time: The device waits a specific amount of time between consecutive SPF calculations. By default, it waits zero seconds. You can configure the SPF hold time to a value from 0 through 65535 seconds. If you set the SPF hold time to 0 seconds, the device does not wait between consecutive SPF calculations.

You can set the SPF delay and hold time to lower values to cause the device to change to alternate paths more quickly if a route fails. Note that lower values for these parameters require more CPU processing time.

You can change one or both of the timers.

> **NOTE**
> If you want to change only one of the timers, for example, the SPF delay timer, you must specify the new value for this timer as well as the current value of the SPF hold timer, which you want to retain. The device does not accept only one timer value.

> **NOTE**
> If you configure SPF timers between 0 through 100, they default to 0.

Modifying Shortest Path First timers

The Shortest Path First (SPF) throttle timers can be modified.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode..

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config)# router ospf
```

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **timers** command with the **throttle spf** keyword and specify the SPF delay, the hold time, and the maximum wait time.

```
device(config-ospf-router)# timers throttle spf 100 500 5000
```

```
device(config-router-ospf-vrf-default-vrf)# timers throttle spf 100 500 5000
```

The following example sets the SPF initial delay to 100 milliseconds, the hold time to 500 milliseconds, and the maximum wait time to 5000 milliseconds.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# timers throttle spf 100 500 5000
```

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# timers throttle spf 100 500 5000
```

```
device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# timers throttle spf 100 500 5000
```

OSPFv2 administrative distance

Devices can learn about networks from various protocols and select a route based on the source of the route information. This decision can be influenced if the default administrative distance for OSPFv2 routes is changed. Consequently, the routes to a network may differ depending on the protocol from which the routes were learned.

You can influence the device's decision by changing the default administrative distance for OSPFv2 routes. You can configure a unique administrative distance for each type of OSPFv2 route. For example, you can configure the Extreme device to prefer a static route over an OSPFv2 inter-area route and to prefer OSPFv2 intra-area routes over static routes. The distance you specify influences the choice of routes when the device has multiple routes to the same network from different protocols. The device prefers the route with the lower administrative distance.

You can specify unique default administrative distances for the following OSPFv2 route types:

- External routes
- Intra-area routes
- Inter-area routes
- Route maps

NOTE
The choice of routes within OSPFv2 is not influenced. For example, an OSPFv2 intra-area route is always preferred over an OSPFv2 inter-area route, even if the intra-area route's distance is greater than the inter-area route's distance.

Configuring administrative distance based on route type

The default administrative distances for intra-area routes, inter-area routes, route maps, and external routes can be altered.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   ```

3. Enter the **distance** command with the **intra-area** parameter.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# distance intra-area 80
   ```

   The administrative distance for intra-area routes is changed from the default to 80.

4. Enter the **distance** command with the **inter-area** parameter.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# distance inter-area 90
   ```

   The administrative distance for inter-area routes is changed from the default to 90.

5. Enter the **distance** command with the **route-map** parameter.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# distance route-map 85
   ```

   The administrative distance for inter-area routes is changed from the default to 85.

6. Enter the **distance** command with the **external** parameter.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# distance external 100
   ```

   The administrative distance for external routes is changed from the default to 100.

The following example changes the default administrative distances for intra-area routes, inter-area routes, and external routes.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# distance intra-area 80
device(config-ipv6-router-ospf-vrf-default-vrf)# distance inter-area 90
device(config-ipv6-router-ospf-vrf-default-vrf)# distance route-map 85
device(config-ipv6-router-ospf-vrf-default-vrf)# distance external 100
```

OSPFv2 LSA refreshes

To prevent a refresh from being performed each time an individual LSA's refresh timer expires, OSPFv2 LSA refreshes are delayed for a specified time interval. This pacing interval can be altered.

The device paces OSPFv2 LSA refreshes by delaying the refreshes for a specified time interval instead of performing a refresh each time an individual LSA's refresh timer expires. The accumulated LSAs constitute a group, which the device refreshes and sends out together in one or more packets.

The pacing interval, which is the interval at which the device refreshes an accumulated group of LSAs, is configurable in a range from 10 through 1800 seconds (30 minutes). The default is 240 seconds (4 minutes). Thus, every four minutes, the device refreshes the group of accumulated LSAs and sends the group together in the same packets.

The pacing interval is inversely proportional to the number of LSAs the device is refreshing and aging. For example, if you have approximately 10,000 LSAs, decreasing the pacing interval enhances performance. If you have a very small database (40 to 100 LSAs), increasing the pacing interval to 10 to 20 minutes may enhance performance only slightly.

Configuring the OSPFv2 LSA pacing interval

The interval between OSPFv2 LSA refreshes can be modified.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID t o enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

   ```
   device(config)# router ospf
   device(config-rbridge-id-122)# router ospf
   ```

4. Enter the **timers** command with the **lsa-group-pacing** parameter.

   ```
   device(config-ospf-router)# timers lsa-group-pacing 120
   device(config-router-ospf-vrf-default-vrf)# timers lsa-group-pacing 120
   ```

   The OSPFv2 LSA pacing interval is changed to 120 seconds (2 minutes).

The following example changes the OSPFv2 LSA pacing interval is changed to 120 seconds (2 minutes).

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# timers lsa-group-pacing 120

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# timers lsa-group-pacing 120

device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# timers lsa-group-pacing 120
```

Support for OSPF RFC 2328 Appendix E

Extreme devices support Appendix E in OSPF RFC 2328. Appendix E describes a method to ensure that an OSPF device generates unique link state IDs for type-5 (External) link state advertisements (LSAs) in cases where two networks have the same network address but different network masks.

> **NOTE**
> Support for Appendix E of RFC 2328 is enabled automatically and cannot be disabled. No user configuration is required.

Normally, an OSPF device uses the network address alone for the link state ID of the link state advertisement (LSA) for the network. For example, if the device needs to generate an LSA for network 10.1.2.3 255.0.0.0, the device generates ID 10.1.2.3 for the LSA.

However, suppose that an OSPF device needs to generate LSAs for all the following networks:

- 10.0.0.0 255.0.0.0
- 10.0.0.0 255.255.0.0
- 10.0.0.0 255.255.255.0

All three networks have the same network address, 10.0.0.0. Without support for RFC 2328 Appendix E, an OSPF device uses the same link state ID, 10.0.0.0, for the LSAs for all three networks. For example, if the device generates an LSA with ID 10.0.0.0 for network 10.0.0.0 255.0.0.0, this LSA conflicts with the LSA generated for network 10.0.0.0 255.255.0.0 or 10.0.0.0 255.255.255.0. The result is multiple LSAs that have the same ID but that contain different route information.

When appendix E is supported, the device generates the link state ID for a network as the following steps.

1. Does an LSA with the network address as its ID already exist?
   - – No - Use the network address as the ID.
     – Yes - Go to "Support for OSPF RFC 2328 Appendix E".

2. Compare the networks that have the same network address, to determine which network is more specific. The more specific network is the one that has more contiguous one bits in its network mask. For example, network 10.0.0.0 255.255.0.0 is more specific than network 10.0.0.0 255.0.0.0, because the first network has 16 ones bits (255.255.0.0) whereas the second network has only 8 ones bits (255.0.0.0).
   - – For the less specific network, use the networks address as the ID.
     – For the more specific network, use the network's broadcast address as the ID. The broadcast address is the network address, with all ones bits in the host portion of the address. For example, the broadcast address for network 10.0.0.0 255.255.0.0 is 10.0.255.255.

   If this comparison results in a change to the ID of an LSA that has already been generated, the device generates a new LSA to replace the previous one. For example, if the device has already generated an LSA for network with ID 10.0.0.0 for network 10.0.0.0 255.255.255.0, the device must generate a new LSA for the network, if the device needs to generate an LSA for network 10.0.0.0 255.255.0.0 or 10.0.0.0 255.0.0.0.

OSPFv2 graceful restart

The graceful restart (GR) feature provides a routing device with the capability to inform its neighbors when it is performing a restart.

NOTE
IS-IS GR is supported only on the SLX 9850 with redundant MMs.

Neighboring devices, known as GR helpers, are informed via protocol extensions that the device is undergoing a restart and assist in the restart. For the duration of the graceful restart, the restarting device and its neighbors continue forwarding packets ensuring there is no disruption to network performance or topology. Disruptions in forwarding are minimized and route flapping diminished. When the restart is complete, the device is able to quickly resume full operation due to the assistance of the GR helpers. The adjacent devices then return to normal operation.

There are two types of OSPFv2 graceful restart:

- **Planned restart**: The restarting routing device informs its neighbors before performing the restart. The GR helpers act as if the routing device is still within the network topology, continuing to forward traffic to the restarting routing device. A defined interval, known as a "grace period" is set to specify when the neighbors should consider the restart complete and the restarting routing device as part of the network topology again.
- **Unplanned restart**: The routing device restarts without warning due to a software fault.

NOTE
In order for a graceful restart on a routing device to be successful, the OSPFv2 neighbors must have GR-helper mode enabled. GR-helper mode is enabled by default.

NOTE
Process restart takes precedence over GR or non-stop routing (NSR).

OSPFv2 Graceful Restart can be enabled in the following configurations:

- Configuring OSPFv2 Graceful Restart for the Global Instance – In this configuration all OSPFv2 neighbors other than those used by VRFs are made subject to the Graceful Restart capability. The restart timer set globally does not apply to Graceful Restart on a configured VRF.
- Configuring OSPFv2 Graceful Restart per VRF – In this configuration all OSPFv2 neighbors for the specified VRF are made subject to the Graceful Restart capability. The restart timer set for a specific VRF only applies to that VRF.

NOTE
If a 32-slot XMR Series or MLX Series system running a version 03.6.00 or later application image is configured for OSPFv2 graceful restart and intended to be used in switchover or hitless upgrade, the OSPFv2 dead-interval needs to be changed to 60 seconds on OSPFv2 interfaces to ensure that the graceful restart process succeeds without a timeout.

Hitless upgrade support for OSPF graceful restart

OSPFv2 graceful restart experiences minimal packet loss during hitless upgrade on a non-default VRF. On a default VRF, there is no packet loss during hitless upgrade.

Disabling OSPFv2 graceful restart

OSPFv2 graceful restart (GR) is enabled by default, and can be disabled on a routing device.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config)# router ospf

device(config-rbridge-id-122)# router ospf
```

4. Enter the **no graceful restart** command to disable GR on the device.

```
device(config-ospf-router)# no graceful-restart

device(config-router-ospf-vrf-default-vrf)# no graceful-restart
```

The following example disables GR.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# no graceful-restart

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# no graceful-restart

device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# no graceful-restart
```

Re-enabling OSPFv2 graceful restart

If you disable OSPFv2 graceful restart (GR), you can re-enable it. You can also change the maximum restart wait time from the default value of 120 seconds.

NOTE
GR is mutually exclusive to NSR.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config)# router ospf
```

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **graceful restart** command to re-enable GR on the device.

```
device(config-ospf-router)# graceful-restart
```

```
device(config-router-ospf-vrf-default-vrf)# graceful-restart
```

5. Enter the **graceful restart** command with the **restart-time** parameter and specify a value to change the maximum restart wait time from the default value of 120 seconds.

```
device(config-ospf-router)# graceful-restart restart-time 240
```

```
device(config-router-ospf-vrf-default-vrf)# graceful-restart restart-time 240
```

The following example re-enables GR and changes the maximum restart wait time from the default value of 120 seconds to 240 seconds.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# graceful-restart
device(config-ospf-router)# graceful-restart restart-time 240

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# graceful-restart
device(config-router-ospf-vrf-default-vrf)# graceful-restart restart-time 240

device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# graceful-restart
device(config-router-ospf-vrf-default-vrf)# graceful-restart restart-time 240
```

Disabling OSPFv2 graceful restart helper

The OSPFv2 graceful restart (GR) helper is enabled by default, and can be disabled on a routing device.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **graceful-restart** command using the **helper-disable** keyword to disable the GR helper.

```
device(config-ospf-router)# graceful-restart helper-disable
```

```
device(config-router-ospf-vrf-default-vrf)# graceful-restart helper-disable
```

The following example disables the GR helper.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# graceful-restart helper-disable

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# graceful-restart helper-disable

device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# graceful-restart helper-disable
```

OSPFv2 stub router advertisement

OSPFv2 stub router advertisement is an open standard based feature and it is specified in RFC 3137. This feature provides a user with the ability to gracefully introduce and remove an OSPFv2 device from the network, by controlling when the data traffic can start and stop flowing through the device in cases where there are other OSPFv2 devices present on the network providing alternative paths for the traffic. This feature does not work if there is no alternative for the traffic through other OSPFv2 routers. The device can control the data traffic flowing through it by changing the cost of the paths passing through the configured device. By setting the path cost high the traffic will be redirected to other OSPFv2 devices providing a lower cost path. This change in path cost is accomplished by setting the metric of the links advertised in the Router LSA to a maximum value. When the OSPFv2 device is ready to forward the traffic, the links are advertised with the real metric value instead of the maximum value.

OSPFv2 stub router advertisement is useful for avoiding a loss of traffic during short periods when adjacency failures are detected and traffic is rerouted. Using this feature, traffic can be rerouted before an adjacency failure occurs due to common services interruptions such as a router being shutdown for maintenance.

OSPFv2 stub router advertisement is also useful during startup because it gives the device enough time to build up its routing table before forwarding traffic. This can be useful where BGP is enabled on the device because it takes time for the BGP routing table to converge.

You can also configure and set a metric value for the following LSA types:

- Summary (type 3 and type 4)
- External (type 5 and type 7)
- Opaque (type 10, TE link)

OSPFv2 Shortest Path First throttling

Rapid triggering of SPF calculations with exponential back-off to offer the advantages of rapid convergence without sacrificing stability. As the delay increases, multiple topology changes can occur within a single SPF. This dampens network activity due to frequent topology changes.

This scheduling method starts with an initial value after which a configured delay time is followed. If a topology change event occurs the SPF is schedule after the time specified by the initial value, the device starts a timer for the time period specified by a configured hold time value. If no topology events occur during this hold time, the router returns to using the initial delay time.

If a topology event occurs during the hold time period, the next hold time period is recalculated to a value that is double the initial value. If no topology events occur during this extended hold time, the device resets to its initial value. If an event occurs during this extended hold time, the next hold time is doubled again. The doubling occurs as long as topology events occur during the calculated hold times until a configured maximum delay time value is reached or no event occurs (which resets the router to the initial hold time). The maximum value is then held until the hold time expires without a topology change event occurring. At any time that a hold time expires without a topology change event occurring, the router reverts to the initial hold value and begins the process all over again.

For example, if you set the initial delay timer to 100 milliseconds, the hold timer to 300 and the maximum hold timer to 2000 milliseconds, the following will occur:

If a topology change occurs the initial delay of 100 milliseconds will be observed. If a topology change occurs during the hold time of 300 milliseconds the hold time is doubled to 600 milliseconds. If a topology change event occurs during the 600 millisecond period, the hold time is doubled again to 1200 milliseconds. If a topology change event occurs during the 1200 millisecond period, the hold time is doubled to 2400 milliseconds. Because the maximum hold time is specified as 2000, the value will be held at 2000. This 2000 millisecond period will then repeat as long as topology events occur within the maximum 2000 millisecond hold time. When a maximum hold time expires without a topology event occurring, the router reverts to the initial delay time and the cycle repeats as described.

Therefore, longer SPF scheduling values can be used during network topology instability.

IETF RFC and internet draft support

The implementation of OSPF Graceful Restart supports the following IETF RFC:

- RFC 3623: Graceful OSPF Restart

  NOTE
  A secondary management module must be installed for the device to function as a graceful restart device. If the device functions as a graceful restart helper device only, there is no requirement for a secondary management module.

OSPFv2 non-stop routing

OSPFv2 can continue operation without interruption during hitless failover when the OSPFv2 non-stop routing (NSR) feature is enabled.

NOTE
IS-IS NSR is supported only on the SLX 9850 with redundant MMs.

During graceful restart (GR), the restarting neighbors must help build routing information during a failover. However, GR may not be supported by all devices in a network. NSR eliminates this dependency.

NSR does not require support from neighboring devices to perform hitless failover, and OSPF can continue operation without interruption.

NOTE
NSR does not support IPv6-over-IPv4 tunneling and virtual links, so traffic loss is expected while performing hitless failover.

NOTE
NSR does not support IPv6-over-IPv4 tunnel and virtual links, so traffic loss is expected while performing hitless failover.

NOTE
NSR does not support virtual links, so traffic loss is expected while performing hitless failover.

If the active management module fails, the standby management module takes over and maintains the current OSPF routes, link-state advertisements (LSAs), and neighbor adjacencies, so that there is no loss of existing traffic to the OSPF destination.

NOTE
NSR and Graceful Restart (GR) are mutually exclusive.

NOTE
Process restart takes precedence over GR or non-stop routing (NSR).

Limitations of NSR

- Configurations that occur before the switchover are lost due to the CLI synchronization.
- Sham links are not supported.
- OSPF adjacency over GRE tunnels is not supported.
- Changes in the neighbor state or interface state before or during a switchover do not take effect.
- Traffic counters are not synchronized because the neighbor and LSA database counters are recalculated on the standby module during synchronization.
- LSA acknowledging is delayed because it has to wait until standby acknowledging occurs.
- Depending on the sequence of redistribution or new LSAs (from neighbors), the LSAs accepted within the limits of the database may change after switchover.
- In NSR hitless failover, after switchover, additional flooding-related protocol traffic is generated to the directly connected neighbors.
- OSPF startup timers, database overflow, and max-metric, are not applied during NSR switchover.
- Devices may generate OSPF log messages or reset OSPF neighbor timers, but these issues do not cause any OSPF or traffic disruption.

BFD with OSPF NSR

Bidirectional forwarding detection (BFD) supports MP switchover and all BFD sessions for OSPF with graceful OSPF NSR, which are in the up state after the switchover. The BFD sessions for OSPF that do not use OSPF NSR are cleared before the switchover and then re-established on the new active MP after the MP switchover.

In case the active MP learns an OSPF neighbor and then restarts before a new BFD session is established, the standby module will not have a BFD session for the new OSPF neighbor. To overcome this and to support OSPF NSR with BFD, the following functions are supported when the active MP restarts:

- During MP switchover, BFD checks whether OSPF NSR is enabled. If OSPF NSR is enabled, the existing BFD sessions for OSPF is maintained during the switchover.
- OSPF sets up or clears the BFD sessions after OSPF neighbor transition.
- After the switchover, BFD sessions correspond with the active OSPF neighbor.

Enabling OSPFv2 NSR

OSPFv2 non-stop routing (NSR) can be re-enabled if it has been disabled. The following task re-enables NSR for OSPFv2.

NOTE
GR is mutually exclusive to NSR.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config)# router ospf
```

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **nonstop-routing** command to re-enable NSR on the device.

```
device(config-ospf-router)# nonstop-routing
```

```
device(config-router-ospf-vrf-default-vrf)# nonstop-routing
```

The following example re-enables NSR for OSPFv2.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# nonstop-routing

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# nonstop-routing

device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# nonstop-routing
```

Synchronization of critical OSPFv2 elements

All types of LSAs and the neighbor information are synchronized to the standby module using the NSR synchronization library and IPC mechanism to transmit and receive packets.

Link state database synchronization

To ensure non-stop routing, when the active management module fails the standby management module takes over from the active management module, with the identical OSPF link state database it had before the failure. The next shortest path first (SPF) run after the switchover yields the same result in routes as the active module had before the failure. The OSPF protocol requires that all devices in the network have identical databases.

LSA delayed acknowledging

When an OSPF device receives LSAs from its neighbor, it acknowledges the LSAs. After the acknowledgement is received, the neighbor removes this device from its retransmission list and stops resending the LSAs.

In the case of NSR, the device fails after receiving the LSA from its neighbor and acknowledges that neighbor upon receipt of an LSA. The LSA synchronization to the standby module is then completed. In this case the standby module, when taking over from the active module, does not have that LSA in its database and the already acknowledged neighbor does not retransmit that LSA. For this reason, the NSR-capable device waits for LSA synchronization of the standby module to complete (Sync-Ack) before acknowledging the neighbor that sent the LSA.

LSA syncing and packing

When the LSA processing is completed on the active management module and the decision is made to install the LSA in its link state database (LSDB), OSPF synchronizes that LSA to the standby module. OSPF checks the current state of the database entry, whether or not it is marked for deletion. After checking the database state, OSPF packs the LSA status and other necessary information needed for direct installation in the standby OSPF LSDB, along with the LSA portion. When the LSA reaches the standby module, OSPF checks the database entry state in the buffer and takes appropriate action, such as adding, overwriting, updating, or deleting the LSA from the LSDB.

Neighbor device synchronization

When the neighbor device is added in the active management module, it is synchronized and added to the standby module. When the neighbor is deleted in the active module, it is synchronized to the standby module and deleted in the standby module. When the neighbor device state becomes 2way or full, the neighbor device is synchronized to the standby module. The following attributes of the neighbor device are synchronized to the standby module:

- Neighbor device ID
- Neighbor device IP address
- Destination device or backup destination device information
- Neighbor state 2way or full
- MD5 information
- Neighbor priority

Synchronization limitations

- If a neighbor device is inactive for 30 seconds, and if the standby module takes over in another 10 seconds, the neighbor device cannot be dropped. The inactivity timer starts again and takes another 40 seconds to drop the neighbor device.
- In standby module, the valid neighbor states are loading, down, 2way, and full. If the active management processor (MP) fails when the neighbor state is loading, the standby module cannot continue from loading, but the standby can continue from 2way and tries to establish adjacency between the neighboring devices.

- The minimum OSPF dead-interval timer value is 40 seconds. When the dead-interval value is configured to less than this minimum value, OSPF NSR cannot be supported.

Interface synchronization

Interface information is synchronized for interfaces such as PTPT, broadcast, and non-broadcast. Interface wait time is not synchronized to the standby module. If an interface waits for 30 seconds to determine the identity of the designated router (DR) or the backup designated router (BDR), and if the standby module takes over, the wait timer starts again and takes another 40 seconds for the interface state to change from waiting to BDR, DR, or DROther.

Standby module operations

The standby management module with OSPF configuration performs the following functions:

Neighbor database

Neighbor information is updated in the standby module based on updates from the active module. Certain neighbor state and interface transitions are synchronized to the standby module. By default, the neighbor timers on the standby module are disabled.

LSA database

The standby module processes LSA synchronization events from the active module and unpacks the LSA synchronization information to directly install it in its LSDB, as the LSA has already been processed on the active module. The information required to install all types of LSAs (and special LSAs such as Grace LSAs) is packed by OSPF on the active module in the synchronization buffer, so that you can directly install LSAs on the standby module without extra processing.

The standby module is not allowed to originate any LSAs of its own. This is to maintain all information consistently from the active module. The active module synchronizes self-originated LSAs to the standby module.

LSA aging is not applicable on the standby module. During synchronization from the active module, the current LSA age is recorded and the new database timestamp is created on the standby module to later derive the LSA age as needed.

When the active module sends the LSAs to the standby module, based on the message, the standby module deletes or updates its LSDB with the latest information.

LSA acknowledging or flooding are not done on the standby module. When the LSA synchronization update arrives from the active module, it will be directly installed into the LSDB.

OSPFv2 distribute list

A distribution list can be configured to explicitly deny specific routes from being eligible for installation in the IP route table. By default, all OSPFv2 routes in the OSPFv2 route table are eligible for installation in the IP route table. Receipt of LSAs are not blocked for the denied routes. The device still receives the routes and installs them in the OSPFv2 database. The denied OSPFv2 routes cannot be installed into the IP route table.

The OSPFv2 distribution list can be managed using ACLs or route maps to identify routes to be denied as described in the following sections:

- Configuring an OSPFv2 Distribution List using ACLs

- Configuring an OSPFv2 Distribution List using route maps

Configuring an OSPFv2 distribution list using ACLs

To configure an OSPFv2 distribution list using ACLs:

- Configure an ACL that identifies the routes you want to deny. Using a standard ACL allows you deny routes based on the destination network, but does not filter based on the network mask. To also filter based on the network mask of the destination network, use an extended ACL.
- Configure an OSPFv2 distribution list that uses the ACL as input.

Examples

In the following configuration example, the first three commands configure a standard ACL that denies routes to any 10.x.x.x destination network and allows all other routes for eligibility to be installed in the IP route table. The last three commands change the CLI to the OSPFv2 configuration level and configure an OSPFv2 distribution list that uses the ACL as input. The distribution list prevents routes to any 10.x.x.x destination network from entering the IP route table. The distribution list does not prevent the routes from entering the OSPFv2 database.

```
device(config)# ip access-list standard no_ip
device(config-std-nacl)# deny 10.0.0.0 0.255.255.255
device(config-std-nacl)# permit any
device(config)# router ospf
device(config-ospf-router) # area 0
device(config-ospf-router) # distribute-list no_ip in
```

In the following example, the first three commands configure an extended ACL that denies routes to any 10.31.39.x destination network and allows all other routes for eligibility to be installed in the IP route table. The last three commands change the CLI to the OSPFv2 configuration level and configure an OSPFv2 distribution list that uses the ACL as input. The distribution list prevents routes to any 10.31.39.x destination network from entering the IP route table. The distribution list does not prevent the routes from entering the OSPFv2 database.

```
device(config)# ip access-list extended DenyNet39
device(config-ext-nacl)# deny ip 10.31.39.0 0.0.0.255 any
device(config-ext-nacl)# permit ip any any
device(config)# router ospf
device(config-ospf-router) # area 0
device(config-ospf-router) # distribute-list DenyNet39 in
```

In the following example, the first command configures a numbered ACL that denies routes to any 10.31.39.x destination network and allows all other routes for eligibility to be installed in the IP route table. The last three commands change the CLI to the OSPFv2 configuration level and configure an OSPF distribution list that uses the ACL as input. The distribution list prevents routes to any 10.31.39.x destination network from entering the IP route table. The distribution list does not prevent the routes from entering the OSPFv2 database.

```
device(config)# ip access-list 100 deny ip 10.31.39.0 0.0.0.255 any
device(config)# ip access-list 100 permit ip any any
device(config)# router ospf
device(config-ospf-router) # area 0
device(config-ospf-router) # distribute-list 100 in
```

Configuring an OSPFv2 distribution list using route maps

You can manage an OSPFv2 distribution list using route maps that apply match operations as defined by an ACL or an IP prefix list. You can also use other options available within the route maps and ACLs to further control the contents of the routes that OSPFv2 provides to the IP route table. This section describes an example of an OSPFv2 distribution list using a route map to specify an OSPFv2 administrative distance for routes identified by an IP prefix list.

To configure an OSPFv2 distribution list using route maps:

- Configure a route map that identifies the routes you want to manage
- Optionally configure an OSPFv2 administrative distance to apply to the OSPFv2 routes
- Configure an OSPFv2 distribution list that uses the route map as input

In the following example, the first two commands identify two routes using the **ip prefix-list test1** command. Next, a route map is created using the **prefix-list test1** command to identify the two routes and the **set distance** command to set the OSPFv2 administrative distance of those routes to 200. A distribution list is then configured under the OSPFv2 configuration that uses the route map named "setdistance" as input.

```
device(config)# ip prefix-list test1 seq 5 permit 10.100.1.0/24
device(config)# ip prefix-list test1 seq 10 permit 10.100.2.0/24
device(config)# route-map setdistance permit 1
device(config-routemap setdistance)# match ip address prefix-list test1
device(config-routemap setdistance)# set distance 200
device(config-routemap setdistance)# exit
device(config)# route-map setdistance permit 2
device(config-routemap setdistance)# exit
device(config)# router ospf
device(config-ospf-router)# area 0
device(config-ospf-router)# area 1
device(config-ospf-router)# distribute-list route-map setdistance in
device(config-ospf-router)# exit
```

Once this configuration is implemented, the routes identified by the **ip prefix-list** command and matched in the route map will have their OSPFv2 administrative distance set to 200. This is displayed in the output from the **show ip route** command, as shown below.

```
device# show ip route
Total number of IP routes: 4
Type Codes - B:BGP D:Connected I:ISIS S:Static R:RIP O:OSPF; Cost - Dist/Metric
      Destination        Gateway     Port       Cost          Type
1     10.1.1.0/24        DIRECT      eth 1/1    0/0                     D
2     10.100.1.0/24      10.1.1.1    eth 1/1    200/2       O
3     10.100.2.0/24      10.1.1.1    eth 1/1    200/10      02
4     10.100.6.0/24      10.1.1.1    eth 1/1    110/2       O
```

Routes 1 and 2 demonstrate the actions of the example configuration as both display an OSPFv2 administrative distance value of 200. Note that the value is applied to both OSPFv2 learned routes that match the route-map instance containing the set distance clause. The other OSPFv2 route (route 3), which does not match the relevant instance, continues to have the default OSPFv2 administrative distance of 110.

OSPFv2 route redistribution

Route redistribution imports and translates different protocol routes into a specified protocol type. On the device, redistribution is supported for static routes, ISIS, OSPF, RIP, and BGP. OSPF redistribution supports the import of static, ISIS, RIP, and BGP routes into OSPF routes.

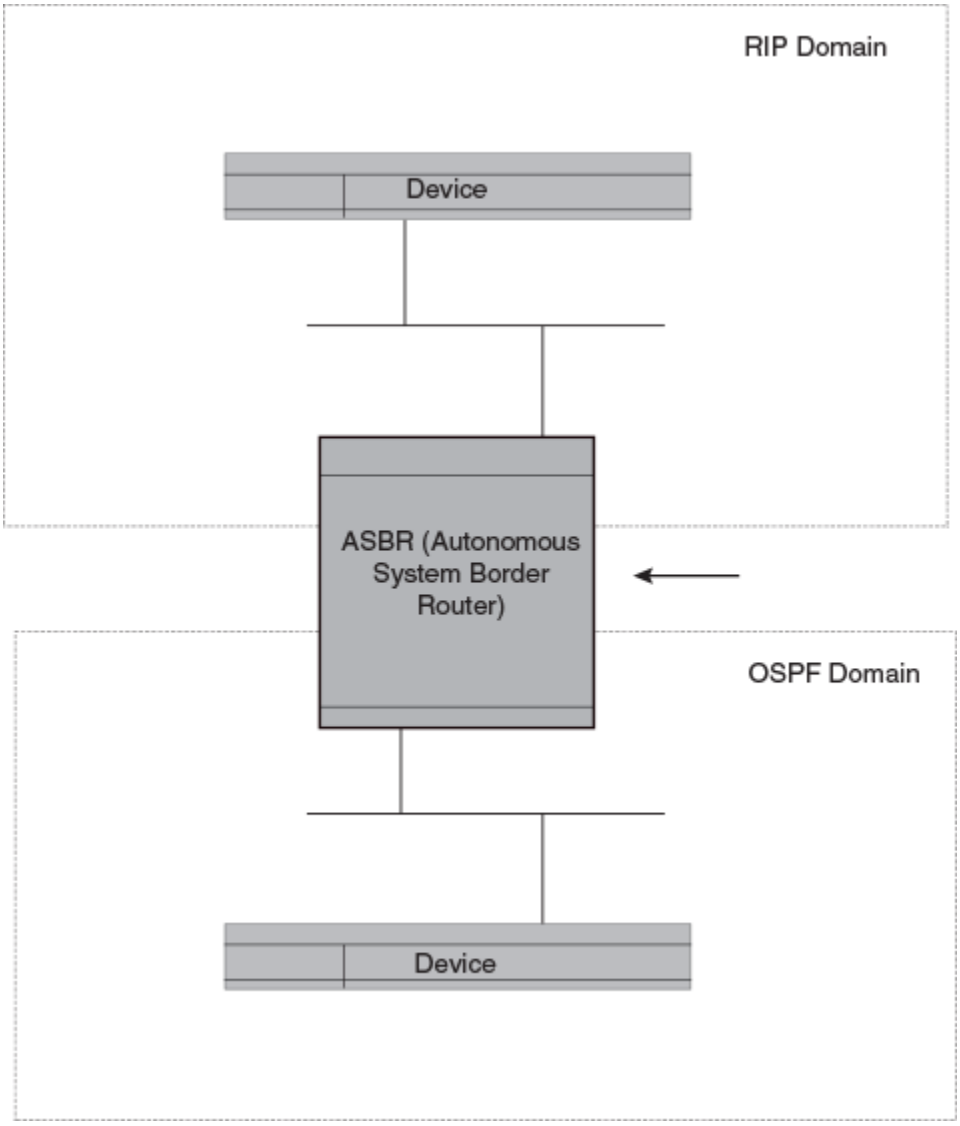
NOTE
The device advertises the default route into OSPF even if redistribution is not enabled, and even if the default route is learned through an IBGP neighbor. IBGP routes (including the default route) are not redistributed into OSPF by OSPF redistribution (for example, by the OSPF **redistribute** command).

In the figure below, the device acting as the ASBR (Autonomous System Boundary Router) can be configured between the RIP domain and the OSPF domain to redistribute routes between the two domains.

NOTE
The ASBR must be running both RIP and OSPF protocols to support this activity.

NOTE
Do not enable redistribution until you have configured the redistribution route map. Otherwise, you might accidentally overload the network with routes you did not intend to redistribute.

FIGURE 73 Redistributing OSPF and static routes to RIP routes



Redistributing routes into OSPFv2

OSPFv2 routes can be redistributed, and the routes to be redistributed can be specified.

The redistribution of RIP and static IP routes into OSPFv2 is configured on a device.

The redistribution of BGP, connected, and static IP routes into OSPFv2 is configured on a device.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **router ospf** command to enter OSPFv2 router configuration mode and enable OSPFv2 on the device.

   ```
   device(config)# router ospf
   ```

3. Enter the **redistribute** command with the **static** parameter to redistribute static routes.

   ```
   device(config-ospf-router)# redistribute static
   device(config-router-ospf-vrf-default-vrf)# redistribute static
   ```

4. Enter the **redistribute** command with the **connected** parameter to redistribute static routes.

   ```
   device(config-router-ospf-vrf-default-vrf)# redistribute connected
   ```

5. Enter the **redistribute** command with the **rip** parameter to redistribute RIP routes.

   ```
   device(config-ospf-router)# redistribute rip
   ```

6. Enter the **redistribute** command with the **bgp** parameter to redistribute BGP routes.

   ```
   device(config-router-ospf-vrf-default-vrf)# redistribute bgp
   ```

The following example redistributes static and RIP routes into OSPFv2 on a device.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# redistribute static
device(config-ospf-router)# redistribute rip
```

The following example redistributes static and BGP routes into OSPFv2 on a device.

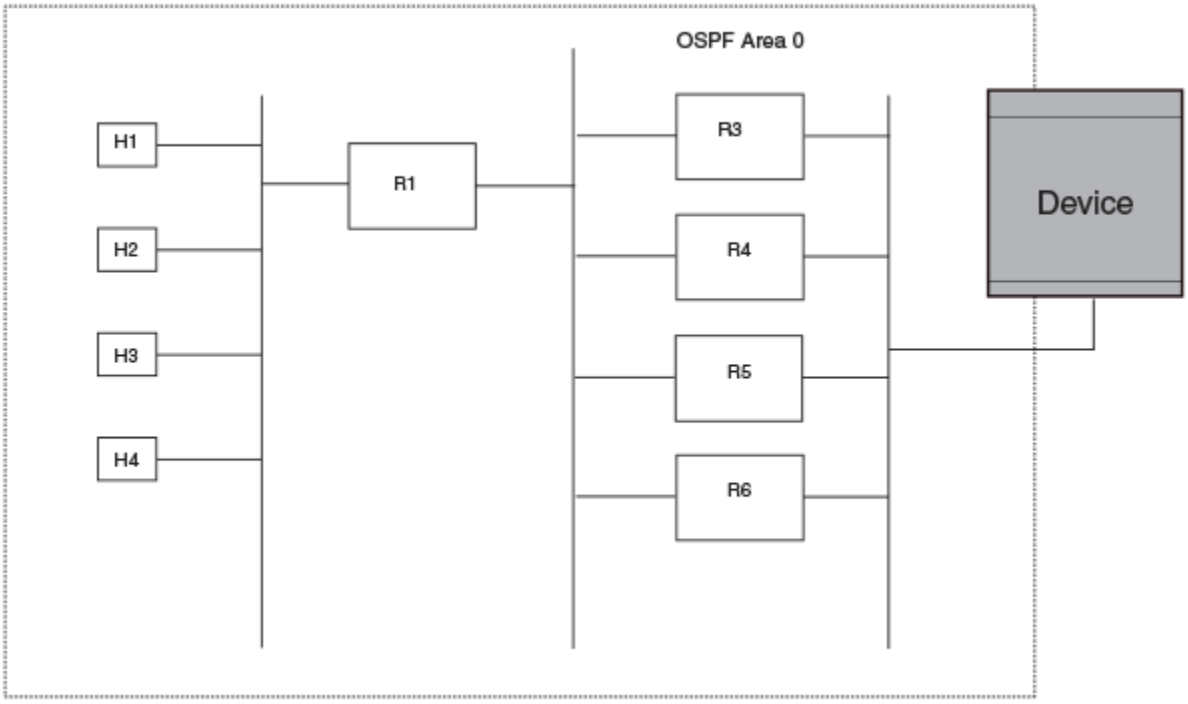
```
device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# redistribute static
device(config-router-ospf-vrf-default-vrf)# redistribute connected
device(config-router-ospf-vrf-default-vrf)# redistribute bgp
```

Load sharing

Extreme devices can load share among up to eight equal-cost IP routes to a destination. By default, IP load sharing is enabled. The default is 4 equal-cost paths but you can specify from 2 to 8 paths.

The device software can use the route information it learns through OSPF to determine the paths and costs.

FIGURE 74 Example OSPF network with four equal-cost paths



The device has four paths to R1:

- Router ->R3
- Router ->R4
- Router ->R5
- Router ->R6

Normally, the device chooses the path to the R1 with the lower metric. For example, if the metric for R3 is 1400 and the metric for R4 is 600, the device always chooses R4.

However, suppose the metric is the same for all four routers in this example. If the costs are the same, the device now has four equal-cost paths to R1. To allow the device to load share among the equal cost routes, enable IP load sharing. Four equal-cost OSPF paths are supported by default when you enable load sharing.

> **NOTE**
> The device is not source routing in these examples. The device is concerned only with the paths to the next-hop routers, not the entire paths to the destination hosts.

OSPF load sharing is enabled by default when IP load sharing is enabled.

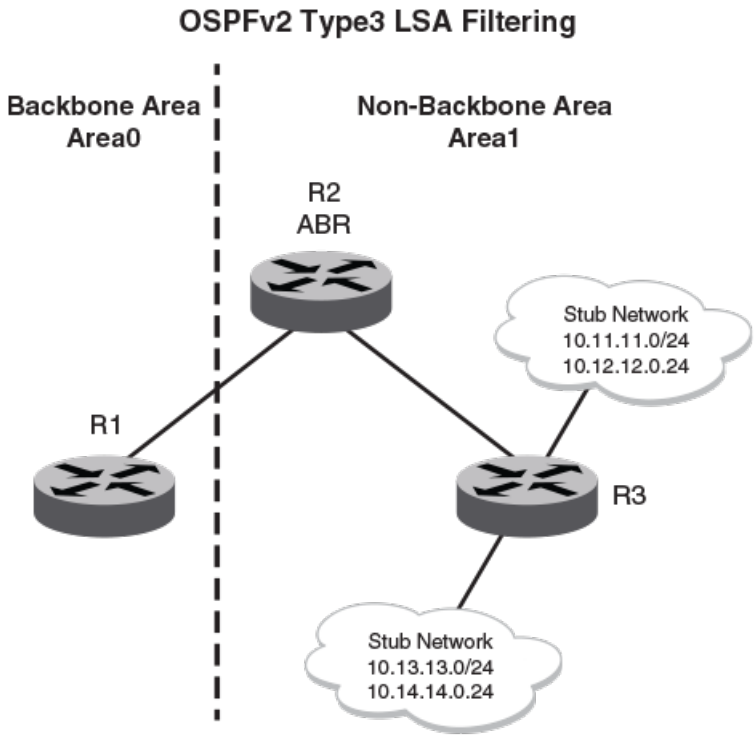
OSPFv2 type 3 LSA filtering

OSPFv2 type 3 LSA filtering provides an ABR that is running the OSPFv2 protocol with the ability to filter type 3 link-state advertisements (LSAs) that are sent between different OSPFv2 areas. Filtering of routes can be defined using prefix-list filters to either permit or deny certain prefixes. Only specified prefixes can be sent from one area to another area and all other prefixes are prohibited. OSPFv2 type 3 LSA filtering can be applied for the LSAs coming into a specific OSPFv2 area or going out of a specific OSPFv2 area, or into and out of the same OSPFv2 areas concurrently. Any change in the prefix-list used for type 3 filtering may result in the advertisement of new summary LSAs or the withdrawal of previously advertised summary LSAs.

Type 3 LSAs refer to summary links and are sent by ABRs to advertise destinations outside the area. OSPFv2 type 3 LSA filtering gives the administrator improved control of route distribution between OSPFv2 areas.

In certain situations, reachability for some IP network prefixes from outside of an area or to a specific area should be restricted. Such a situation is illustrated in the figure below where a device called R3 is advertising stub networks that belong to the non-backbone area (Area1). An ABR, R2, will generate summary routes for these prefixes. If OSPFv2 type 3 LSA filtering is not configured, all the stub networks are seen as summary LSAs in the backbone area (Area 0) and are flooded to all applicable OSPFv2 devices. These type 3 LSAs can be filtered out using the OSPFv2 type3 LSA Filter.

FIGURE 75 OSPFv2 type 3 LSA filtering



Usage and configuration guidelines

- OSPFv2 type 3 LSA filtering is only applicable to ABRs. Configurations are accepted prior to the device becoming an ABR but OSPFv2 type 3 LSA filtering only occurs once the device becomes an ABR.
- When OSPFv2 type 3 LSA filtering is enabled in the “in” direction, all type 3 LSAs originated by the ABR to this area are filtered by the prefix list, based on information from all other areas. Type 3 LSAs, originated as a result of the **area range** command in a

different area, are treated like any other individuallly originated type 3 LSA. Any prefix that does not match an entry in the prefix list is implicitly denied.

- When OSPFv2 type 3 LSA filtering is enabled in the "out" direction, all type 3 LSAs advertised by the ABR are filtered by the prefix list, based on information from this area to all other areas. If the **area range** command has been used to configure type 3 LSAs for this area, these type 3 LSAs that correspond to these configurations are treated like any other type 3 LSA. Prefixes that do not match are implicitly denied.
- Type 3 LSAs received from other devices are not filtered out. The OSPFv2 type 3 LSA filter is only applied when the ABR generates summary routes to advertise.
- If a prefix-list used in type 3 LSA filtering is not created or defined, all summary LSAs are discarded.
- OSPFv2 type 3 LSA filtering is supported for both default VRFs and non-default VRFs.

TABLE 39 Behavior for prefix list configurations

IP prefix list	OSPF area prefix list	Event	Filtering done
XXX	Not defined	None	No (permit all)
Not defined	Defined	None	Yes (deny all)
Not defined	Defined	IP prefix list defined	Recalculation
Defined (no rules configured)	Defined	None	Implicit deny (deny all)
Defined (rules configured)	Defined	IP prefix list deleted	Recalculation and deny all
Defined (rules configured)	Defined	IP prefix list rule added or modified or deleted	Recalculation
Defined (rules configured)	Defined	Area prefix list deleted	Recalculation and permit all

Configuring OSPFv2 type 3 LSA filtering

An IP prefix list can be configured and applied to an OSPFv2 area.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip prefix-list** command and specify a name.

```
device(config)# ip prefix-list mylist permit 10.1.0.0/16
device(config-rbridge-id-122)# ip prefix-list mylist permit 10.1.0.0/16
```

   Configures an IP prefix list named "mylist" that permits routes to network 10.1.0.0/16.

4. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
device(config-rbridge-id-122)# router ospf
```

5. Enter the **area** command to define an OSPFv2 area ID.

```
device(config-ospf-router)# area 1
device(config-router-ospf-vrf-default-vrf)# area 1
```

6. Enter the **area prefix-list** command and specify an area address and a prefix-list.

```
device(config-ospf-router)# area 1 prefix-list mylist out
```

```
device(config-router-ospf-vrf-default-vrf)# area 1 prefix-list mylist out
```

Configures the device to use the IP prefix list “mylist” to determine which routes from Area 1 are advertised to other areas. The device advertises routes that fall within the 10.1.0.0/16 range to other areas because the IP prefix list explicitly permits these routes to be sent to other areas from Area 1.

The following example onfigures an IP prefix list named “mylist” that permits routes to network 10.1.0.0/16. This list is used to determine which routes to send to Area 1.

```
device# configure terminal
device(config)# ip prefix-list mylist permit 10.1.0.0/16
device(config)# router ospf
device(config-ospf-router)# area 1
device(config-ospf-router)# area 1 prefix-list mylist out

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip prefix-list mylist permit 10.1.0.0/16
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 1
device(config-router-ospf-vrf-default-vrf)# area 1 prefix-list mylist out

device# configure terminal
device(config)# ip prefix-list mylist permit 10.1.0.0/16
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 1
device(config-router-ospf-vrf-default-vrf)# area 1 prefix-list mylist out
```

Verifying OSPFv2 type 3 LSA filtering configurations

Use this example procedure to verify OSPFv2 type 3 LSA filtering configurations.

1. Verify OSPFv2 configurations at any level of the CLI.

```
device# show ip ospf database link-state

Link States

Index Area ID          Type LS ID           Adv Rtr          Seq(Hex)         Age  Cksum
1     0                Rtr  192.0.0.1       192.0.0.1        0x80000002       404  0x60b6
2     0                Rtr  1.1.1.1         1.1.1.1          0x80000002       403  0xa9e8
3     0                Net  30.30.30.1      1.1.1.1          0x80000001       403  0x959f
4     0                Summ 10.14.14.0      1.1.1.1          0x80000001       211  0x9b95
5     0                Summ 10.13.13.0      1.1.1.1          0x80000001       237  0xb280
6     0                Summ 10.12.12.0      1.1.1.1          0x80000001       264  0xc96b
7     0                Summ 10.11.11.0      1.1.1.1          0x80000001       542  0xe056
8     0                Summ 20.20.20.0      1.1.1.1          0x80000001       542  0x8497
9     1                Rtr  3.3.3.3         3.3.3.3          0x80000006       212  0x7a6e
10    1                Rtr  1.1.1.1         1.1.1.1          0x80000003       542  0x5775
11    1                Net  20.20.20.2      3.3.3.3          0x80000001       1728 0xd827
12    1                Summ 30.30.30.0      1.1.1.1          0x80000001       542  0x1be2
```

Information about all summary LSAs is displayed.

2. Configure type 3 LSA filtering on the device.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 0 prefix-list AREA_0_IN in
device(config-router-ospf-vrf-default-vrf)# exit
device(config-rbridge-id-122)# ip prefix-list AREA_0_IN  seq 20 deny 10.12.12.0/24
device(config-rbridge-id-122)# ip prefix-list AREA_0_IN  seq 30 permit 10.14.14.0/24
```

   The following filters are applied:

   - Area 0: Inbound filtering permits 10.14.14.0/24 and denies 10.12.12.0/24.4

3. Verify OSPFv2 type 3 LSA filtering configurations using the **show ip ospf database link-state** command.

```
device# show ip ospf database link-state

Link States

Index Area ID         Type LS ID           Adv Rtr         Seq(Hex)        Age  Cksum
1     0               Rtr  192.0.0.1       192.0.0.1       0x80000002      1478 0x60b6
2     0               Rtr  1.1.1.1         1.1.1.1         0x80000002      1477 0xa9e8
3     0               Net  30.30.30.1      1.1.1.1         0x80000001      1477 0x959f
4     0               Summ 10.14.14.0      1.1.1.1         0x80000001      1285 0x9b95
5     1               Rtr  3.3.3.3         3.3.3.3         0x80000006      1286 0x7a6e
6     1               Rtr  1.1.1.1         1.1.1.1         0x80000003      1616 0x5775
7     1               Net  20.20.20.2      3.3.3.3         0x80000002      938  0xd628
8     1               Summ 30.30.30.0      1.1.1.1         0x80000001      1616 0x1be2
```

4. Verify the settings for type 3 LSA filtering in the in direction for OSPFv2 area 0 using the **show ip ospf filtered-lsa area** command.

```
device# show ip ospf filtered-lsa area 0 in

Prefix List Name: AREA_0_IN
Direction: IN
Area: 0
Filtered LSA list:
 Prefix          Mask
10.11.11.0      255.255.255.0
10.12.12.0      255.255.255.0
10.13.13.0      255.255.255.0
20.20.20.0      255.255.255.0
Total number of Filtered LSA :4
```

5. Verify information about type 3 LSA filters configured for OSPFv2 areas using the **show ip ospf config** command.

```
device# show ip ospf config

Router OSPF: Enabled
Nonstop Routing: Disabled
Graceful Restart: Enabled
Graceful Restart Helper: Enabled
Graceful Restart Time: 120

Redistribution: Disabled
Default OSPF Metric: 10
Maximum Paths: 8
OSPF Auto-cost Reference Bandwidth: Disabled
Default Passive Interface: Disabled
OSPF Redistribution Metric: Type2
OSPF External LSA Limit: 14913080
OSPF Database Overflow Interval: 0
RFC 1583 Compatibility: Enabled
VRF Lite capability: Disabled
Router id: 1.1.1.1
OSPF Area currently defined:
Area-ID          Area-Type Cost
1                normal    0
0                normal    0

OSPF type3 filters currently defined:
Area-ID          IN prefix                        OUT prefix
0                AREA_0_IN
```

Interface types to which the reference bandwidth does not apply

Some interface types are not affected by the reference bandwidth and always have the same cost regardless of the reference bandwidth in use:

- The cost of a loopback interface is always 1.
- The cost of a virtual link is calculated using the Shortest Path First (SPF) algorithm and is not affected by the auto-cost feature.
- The bandwidth for tunnel interfaces is 9 Kbps and is also subject to the auto-cost reference bandwidth setting.

Changing the reference bandwidth for the cost on OSPFv2 interfaces

Each interface on which OSPFv2 is enabled has a cost associated with it. The device advertises its interfaces and their costs to OSPFv2 neighbors. For example, if an interface has an OSPFv2 cost of ten, the device advertises the interface with a cost of ten to other OSPFv2 routers.

By default, an interface's OSPFv2 cost is based on the port speed of the interface. The cost is calculated by dividing the reference bandwidth by the port speed. The default reference bandwidth is 100 Mbps, which results in the following default costs:

- 10 Mbps port - 10
- All other port speeds - 1

You can change the reference bandwidth. The following formula is used to calculate the cost:

Cost = reference-bandwidth/interface-speed

If the resulting cost is less than 1, the cost is rounded up to 1. The default reference bandwidth results in the following costs:

- 10 Mbps port's cost = 100/10 = 10
- 100 Mbps port's cost = 100/100 = 1
- 1000 Mbps port's cost = 100/1000 = 0.10, which is rounded up to 1
- 10 Gbps port's cost = 100/10000 = 0.01, which is rounded up to 1

The bandwidth for interfaces that consist of more than one physical port is calculated as follows:

- LAG group - The combined bandwidth of all the ports.
- Virtual interface - The combined bandwidth of all the ports in the port-based VLAN that contains the virtual interface.

The default reference bandwidth is 100 Mbps. You can change the reference bandwidth to a value from 1—4294967.

If a change to the reference bandwidth results in a cost change to an interface, the device sends a link-state update to update the costs of interfaces advertised by the device.

> **NOTE**
> If you specify the cost for an individual interface, the cost you specify overrides the cost calculated by the software.

OSPFv2 over VRF

With Network OS 4.0 and later, OSPFv2 can run over multiple Virtual Routing and Forwarding (VRF) instances. All OSPFv2 commands supported in Network OS 4.0 and later are available over default and non-default OSPF instances.

OSPFv2 can run over multiple Virtual Routing and Forwarding (VRF) instances. All OSPFv2 commands are available over default and non-default OSPF instances.

OSPFv2 maintains multiple instances of the routing protocol to exchange route information among various VRF instances. A multi-VRF-capable device maps an input interface to a unique VRF, based on user configuration. These input interfaces can be physical or a virtual interface. By default, all input interfaces are attached to the default VRF instance.

Multi-VRF for OSPF (also known as VRF-Lite for OSPF) provides a reliable mechanism for trusted VPNs to be built over a shared infrastructure. The ability to maintain multiple virtual routing or forwarding tables allows overlapping private IP addresses to be maintained across VPNs.

Enabling OSPFv2 in a non-default VRF

When OSPFv2 is enabled in a non-default VRF instance, the device enters OSPF router VRF configuration mode. Several commands can then be accessed that allow the configuration of OSPFv2.

A non-default VRF instance has been configured.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command and specify a VRF name to enter OSPF router VRF configuration mode and enable OSPFv2 on a non-default VRF.

```
device(config-rbridge-id-122)# router ospf vrf green
```

```
device(config)# router ospf vrf green
```

The following example enables OSPFv2 in a non-default VRF.

```
device# configure terminal
device(config)# router ospf vrf green
device(config-ospf-router-vrf-green)#

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf vrf green
device(config-router-ospf-vrf-green)#

device# configure terminal
device(config)# router ospf vrf green
device(config-router-ospf-vrf-green)#
```

OSPF VRF-Lite for customer edge routers

When a type 3, 5, or 7 LSA is sent from a provider edge (PE) router to a customer edge (CE) router, the DN (down) bit in the LSA options field must be set. This prevents any type 3, 5, or 7 LSA messages sent from the CE router to the PE router from being distributed any farther. The PE router ignores messages with the DN bit set and does not add these routes to the VRF routing table.

When you enable VRF-Lite on the CE router, the DN setting is ignored, allowing the CE router to add these routes to the VRF routing table.

To enable VRF-Lite, enter commands such as the following:

```
device(config)# router ospf vrf 1
device(config-ospf-router-vrf-1)# vrf-lite-capability
```

Syntax: [no] vrf-lite-capability

Use the **no** form of the command to disable VRF-Lite. This applies to the VRF instance only. It does not apply to the default VRF.

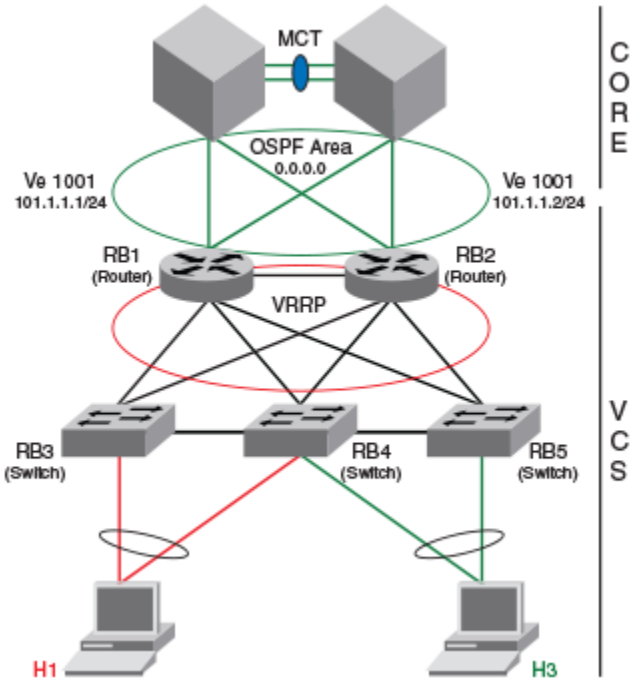
> NOTE
> For vpn4 external routes to be installed on CE routers, the domain-tags on PE routers must be different that the domain-tags on CE routers.

> NOTE
> This command applies to CE routers only. This command does not apply to PE routers.

OSPFv2 in a VCS environment

The figure below shows one way in which OSPFv2 can be used in a VCS Fabric environment. Routers RB1 and RB2, as well as the devices that aggregate traffic to the core, are configured with OSPFv2. Switches RB3, RB4, and RB5 are Layer 2 switches.

FIGURE 76 OSPF example in a VCS environment



OSPFv2 considerations and limitations

- OSPFv2 must be configured in a Virtual Cluster Switching (VCS) environment.
- The following Extreme platforms support OSPF:
  - VDX 6740
  - VDX 6740T
  - VDX 6740T-1G
  - VDX 6940-36Q
  - VDX 6940-144S
  - VDX 8770-4
  - VDX 8770-8
- OSPFv2 can be configured on either a point-to-point or broadcast network.
- OSPF is supported on unnumbered IP interfaces. Refer to the *Unnumbered IP interfaces* section for more information.
- OSPFv2 can be enabled on the following interfaces: gigabitethernet, tengigabitethernet, fortygigabitethernet, loopback, and ve.
- On enabling OSPFv2 over a loopback interface, the network is advertised as a stub network in the router LSA for the attached area. OSPFv2 control packets, such as *hellos*, are not transmitted on loopback interfaces and adjacencies will not form.
- For VXLAN, if you are configuring a loopback interface to serve as a VTEP, you must manually configure distinct router-ids, using the **ip router id command**, for use by routing protocols.

Configuring the OSPFv2 Max-Metric Router LSA

By configuring the OSPFv2 max-metric router LSA you can enable OSPFv2 to advertise its locally generated router LSAs with a maximum metric.

NOTE
You can configure OSPFv2 max-metric router LSA in either startup or non-startup mode. When you configure max-metric in non-startup mode, it only applies once and is not persistent across reloads or after the **clear ip ospf all** command is issued.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config-rbridge-id-122)# router ospf
```

```
device(config)# router ospf
```

4. Enter the **max-metric router-lsa** command with the **on-startup** keyword and specify a value to specify a period of time to advertise a maximum metric after a restart before advertising with a normal metric.

```
device(config-ospf-router)# max-metric router-lsa on-startup 85
```

5. Enter the **max-metric router-lsa** command with the **all-lsas** parameter to set the set the summary-lsa and external-lsa parameters to the corresponding default max-metric value. .

```
device(config-router-ospf-vrf-default-vrf)# max-metric router-lsa all-lsas
```

The following example configures an OSPFv2 device to advertise a maximum metric for 85 seconds after a restart before advertising with a normal metric.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# max-metric router-lsa on-startup 85
```

The following example configures an OSPFv2 device to advertise the maximum metric value using the **all-lsas** parameter.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# max-metric router-lsa all-lsas
```

```
device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# max-metric router-lsa all-lsas
```

Re-enabling OSPFv2 compatibility with RFC 1583

OSPFv2 is compatible with RFC 1583 and maintains a single best route to an autonomous system (AS) boundary router in the OSPF routing table. Disabling this compatibility causes the OSPF routing table to maintain multiple intra-AS paths, which helps prevent routing loops. You can re-enable OSPFv2 compatibility with RFC 1583 if it has been disabled.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ip router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config-rbridge-id-122)# router ospf
   ```

   ```
   device(config)# router ospf
   ```

4. Enter the **rfc1583-compatibility** command to re-enable OSPFv2 compatibility with RFC 1583.

   ```
   device(config-router-ospf-vrf-default-vrf)# rfc1583-compatibility
   ```

The following example re-enables OSPFv2 compatibility with RFC 1583.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# rfc1583-compatibility
```

```
device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# rfc1583-compatibility
```

Enabling OSPFv2 in a VCS environment

Do the following to enable OSPFv2 in a VCS environment.

NOTE

If no RBridge ID is configured on the device, deleting a VE interface will cause a spike in CPU usage. To prevent this, configure an RBridge ID before deleting a VE interface.

1. On device RB1, do the following:
   a) Enter the **configure terminal** command to enter global configuration mode.
   b) Enter the **interface vlan** command followed by the VLAN number to create a VLAN for the router.
   c) Enter the **exit** command to exit interface configuration mode.
   d) Enter the **rbridge-id** command followed by the RBridge ID to enter RBridge configuration mode.
   e) Enter the **router ospf** command to enable the OSPFv2 routing protocol and to enter OSPF router configuration mode.
   f) Enter the **area** command followed by the area ID to create this OSPF area on this router.
   g) Enter the **exit** command to exit OSPF router configuration mode.
   h) Enter the **interface ve** command followed by the VLAN number to enter interface configuration mode.
   i) Enter the **ip address** command followed by the IP address/subnet of the interface.
   j) Enter the **ip ospf area** command followed by the area ID to assign the interface to this area.
   k) Enter the **no shutdown** command.

```
RB1# configure terminal
RB1(config)# interface vlan 1001
RB1(config-Vlan-1001)# exit
RB1(config)# rbridge-id 1

RB1(config-rbridge-id-1)# router ospf
RB1(config-router-ospf-vrf-default-vrf)# area 0.0.0.0
RB1(config-router-ospf-vrf-default-vrf)# exit
RB1(config-rbridge-id-1)# interface ve 1001
RB1(config-Ve-1001)# ip address 101.1.1.1/24
RB1(config-Ve-1001)# ip ospf area 0.0.0.0
RB1(config-Ve-1001)# no shutdown
```

2. On device RB2, do the following:
   a) Enter the **configure terminal** command to enter global configuration mode.
   b) Enter the **interface vlan** command followed by the VLAN number to create a VLAN for the router.
   c) Enter the **exit** command to exit interface configuration mode.
   d) Enter the **rbridge-id** command followed by the RBridge ID to enter RBridge configuration mode.
   e) Enter the **router ospf** command to enable the OSPFv2 routing protocol and to enter OSPF router configuration mode.
   f) Enter the **area** command followed by the area ID to create this OSPF area on this router.
   g) Enter the **exit** command to exit OSPF VRF router configuration mode.
   h) Enter the **interface ve** command followed by the VLAN number to enter interface configuration mode.
   i) Enter the **ip address** command followed by the IP address/subnet of the interface.
   j) Enter the **ip ospf area** command followed by the area ID to assign the interface to this area.
   k) Enter the **no shutdown** command.

```
RB2# configure terminal
RB2(config)# interface vlan 1001
RB2(config-Vlan-1001)# exit
RB2(config)# rbridge-id 2

RB2(config-rbridge-id-2)# router ospf
RB2(config-router-ospf-vrf-default-vrf)# area 0.0.0.0
RB2(config-router-ospf-vrf-default-vrf)# exit
RB2(config-rbridge-id-2)# interface ve 1001
RB2(config-Ve-1001)# ip address 101.1.1.2/24
RB2(config-Ve-1001)# ip ospf area 0.0.0.0
RB2(config-Ve-1001)# no shutdown
```

   l) Assign VLAN 1001 to a vLAG.

Changing default settings

Refer to the relevant Command Reference for other commands you can use to change default OSPF settings. Some commonly configured items include the following:

- Changing reference bandwidth to change interface costs by using the **auto-cost reference-bandwidth** command.
- Defining redistribution filters for the Autonomous System Boundary Router (ASBR) by using the **redistribute** command.

Disabling and re-enabling OSPFv2 event logging

OSPFv2 event logging can be configured, disabled, and re-enabled.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config)# router ospf
```

```
device(config-rbridge-id-1)# router ospf
```

4. Enter the **no log all** command to disable the logging of all OSPFv2 events.

```
device(config-ospf-router)# no log all
```

```
device(config-router-ospf-vrf-default-vrf)# no log all
```

The following example re-enables the logging of all OSPFv2 events.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# log all

device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router ospf
device(config-router-ospf-vrf-default-vrf)# log all

device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# log all
```

Understanding the effects of disabling OSPFv2

Consider the following before disabling OSPFv2 on a device:

- If you disable OSPFv2, the device removes all the configuration information for the disabled protocol from the running configuration. Moreover, when you save the configuration to the running configuration file after disabling one of these protocols, all the configuration information for the disabled protocol is removed from the running configuration file.
- If you have disabled the protocol but have not yet saved the configuration to the running configuration file and reloaded the software, you can restore the configuration information by re-entering the **router ospf** command, or by selecting the Web management option to enable the protocol. If you have already saved the configuration to the running configuration file and reloaded the software, the information is gone.
- If you are testing an OSPFv2 configuration and are likely to disable and re-enable the protocol, you might want to make a backup copy of the running configuration file containing the protocol's configuration information. This way, if you remove the configuration information by saving the configuration after disabling the protocol, you can restore the configuration by copying the backup copy of the startup configuration file into the flash memory.
- If the management default route information is available in the Chassis ID (CID) card, the OSPFv2 default route is overwritten by the management default route when the switch reboots. In order to prevent this, remove the management default route after the switch reboots. The OSPFv2 default route is automatically re-instated. Refer to the "Using the Chassis ID (CID) Recovery Tool "chapter in the *Network OS Software Troubleshooting Guide*.

Disabling OSPFv2

To disable OSPFv2 on a device, use the **no router ospf** command:

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **no router ospf** command to disable OSPFv2 on the device.

```
device(config)# no router ospf

device(config-rbridge-id-122)# no router ospf
```

The following example disables OSPFv2 on a device.

```
device# configure terminal
device(config)# no router ospf

device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# no router ospf
```

Displaying OSPFv2 results

The **show ip ospf** command and its variations can be used to display information about OSPFv2 configurations.

Use one or more of the following commands to verify OSPFv2 information. Using the **show ip ospf** command is optional, and the variations of the command can be entered in any order.

1. In privileged EXEC mode, enter the **show ip ospf** command to display general OSPFv2 information.

```
device# show ip ospf

show ip ospf
OSPF Version                  Version 2
Router Id                     32.32.32.32
ASBR Status                   No
ABR Status                    No         (0)
Redistribute Ext Routes from
Initial SPF schedule delay    0          (msecs)
Minimum hold time for SPFs    0          (msecs)
Maximum hold time for SPFs    0          (msecs)
External LSA Counter          0
External LSA Checksum Sum     0
Originate New LSA Counter     9
Rx New LSA Counter            5
External LSA Limit            14913080
Administrative Distance
  -  External Routes:         110
  -  Intra Area Routes:       110
  -  Inter Area Routes:       110
Database Overflow Interval    0
Database Overflow State :     NOT OVERFLOWED
RFC 1583 Compatibility :      Disabled
NSSA Translator:              Enabled
Nonstop Routing:              Disabled
Graceful Restart              Enabled
Graceful Restart Helper       Enabled
Graceful Restart Time         120
LDP-SYNC: Not globally enabled
Interfaces with LDP-SYNC enabled:
    None
```

The example output displays general OSPFv2 information and indicates that the device is not operating as an ASBR. If the device is not operating as an ASBR, there is no information about redistribution in the output.

2. Enter the **show ip ospf area** command to display information about an OSPFv2 area.

```
device# show ip ospf area

Number of Areas is 1

Index   Area            Type   Cost       SPFR       ABR   ASBR  LSA     Chksum(Hex)
1       0               normal 0          482        0     0     1       00006828
```

The example output displays detailed output for OSPFv2 Area 0.

3. Enter the **show ip ospf interface** command to display OSPFv2 interface information.

```
device# show ip ospf interface

Ethernet 1/8 admin up, oper up
     IP Address 10.1.8.32, Area 0
     BFD is disabled
     Database Filter: Not Configured
     State DR, Pri 1, Cost 1, Options ------E-, Type broadcast Events 8
     Timers(sec): Transmit 1, Retrans 5, Hello 10, Dead 40
     DR:  Router ID 32.32.32.32       Interface Address 10.1.8.32
     BDR: Router ID 0.0.0.0           Interface Address 0.0.0.0
     Neighbor Count = 1, Adjacent Neighbor Count= 0
     Neighbor:         10.1.8.19 [id 19.19.19.18]
     Authentication-Key: None
     MD5 Authentication: Key None, Key-Id  None , Auth-change-wait-time 300
LDP-SYNC: Disabled, State: -

Loopback 1 admin up, oper up
     IP Address 32.32.32.32, Area 0
     BFD is disabled
     Database Filter: Not Configured
     State DR, Pri 1, Cost 1, Options ------E-, Type broadcast Events 2
     Timers(sec): Transmit 1, Retrans 5, Hello 10, Dead 40
     DR:  Router ID 32.32.32.32       Interface Address 32.32.32.32
     BDR: Router ID 0.0.0.0           Interface Address 0.0.0.0
     Neighbor Count = 0, Adjacent Neighbor Count= 0
     Authentication-Key: None
     MD5 Authentication: Key None, Key-Id  None , Auth-change-wait-time 300
LDP-SYNC: Disabled, State: -
```

The example output displays detailed output for OSPFv2 interfaces.

4. Enter the **show ip ospf interface** command with the **brief** parameter to display summarized OSPFv2 interface information.

```
device# show ip ospf interface brief
Interface       Area            IP Addr/Mask         Cost  State    Nbrs(F/C)
Eth 6/45        0               10.10.10.1/24        1     down     0/0
Ve 15           0               15.15.15.1/24        1     DR       0/0
Ve 161          0               161.161.161.1/24     1     DR       0/0
Ve 162          0               162.162.162.1/24     1     DR       0/0
```

The example output displays summarized output for OSPFv2 interfaces.

5. Enter the **show ip ospf neighbor** command to display OSPFv2 neighbor information for the device.

```
device# show ip ospf neighbor

Number of Neighbors is 1, in FULL state 0

Port            Address         Pri State      Neigh Address   Neigh ID        Ev      Opt Cnt
Eth 1/8         10.1.8.32       1   INIT/OTHER 10.1.8.19       19.19.19.18     7       66  0
```

The example output displays output for an OSPFv2 neighbor.

6. Enter the **show ip ospf routes** command to display information about OSPFv2 routes.

```
device# show ip ospf routes

OSPF Regular Routes 7:

      Destination         Mask            Path_Cost Type2_Cost Path_Type
      1.1.1.1             255.255.255.255 1         0          Intra
      Adv_Router          Link_State      Dest_Type State      Tag       Flags
      1.1.1.1             1.1.1.1         Network   Valid      0          6
      Paths Out_Port      Next_Hop        Type      State
      1     Lo 1          0.0.0.0         OSPF      0  0

      Destination         Mask            Path_Cost Type2_Cost Path_Type
      1.1.1.2             255.255.255.255 1         0          Intra
      Adv_Router          Link_State      Dest_Type State      Tag       Flags
      1.1.1.1             1.1.1.1         Network   Valid      0          6
      Paths Out_Port      Next_Hop        Type      State
      1     Lo 2          0.0.0.0         OSPF      0  0

      Destination         Mask            Path_Cost Type2_Cost Path_Type
      1.1.1.3             255.255.255.255 1         0          Intra
      Adv_Router          Link_State      Dest_Type State      Tag       Flags
      1.1.1.1             1.1.1.1         Network   Valid      0          6
      Paths Out_Port      Next_Hop        Type      State
      1     Lo 3          0.0.0.0         OSPF      0  0

      Destination         Mask            Path_Cost Type2_Cost Path_Type
      1.1.1.4             255.255.255.255 1         0          Intra
      Adv_Router          Link_State      Dest_Type State      Tag       Flags
      1.1.1.1             1.1.1.1         Network   Valid      0          6
      Paths Out_Port      Next_Hop        Type      State
      1     Lo 4          0.0.0.0         OSPF      0  0

      Destination         Mask            Path_Cost Type2_Cost Path_Type
      1.1.1.5             255.255.255.255 1         0          Intra
      Adv_Router          Link_State      Dest_Type State      Tag       Flags
      1.1.1.1             1.1.1.1         Network   Valid      0          6
      Paths Out_Port      Next_Hop        Type      State
      1     Lo 5          0.0.0.0         OSPF      0  0

      Destination         Mask            Path_Cost Type2_Cost Path_Type
      1.1.1.6             255.255.255.255 1         0          Intra
      Adv_Router          Link_State      Dest_Type State      Tag       Flags
      1.1.1.1             1.1.1.1         Network   Valid      0          6
      Paths Out_Port      Next_Hop        Type      State
      1     Lo 6          0.0.0.0         OSPF      0  0

      Destination         Mask            Path_Cost Type2_Cost Path_Type
      1.1.1.7             255.255.255.255 1         0          Intra
      Adv_Router          Link_State      Dest_Type State      Tag       Flags
      1.1.1.1             1.1.1.1         Network   Valid      0          6
      Paths Out_Port      Next_Hop        Type      State
      1     Lo 7          0.0.0.0         OSPF      0  0
```

7. Enter the **show ip ospf summary** command to display summary output for OSPFv2 sessions.

```
device# show ip ospf summary

Seq Instance        Intfs   Nbrs    Nbrs-Full LSAs    Routes
1   default-vrf     2       1       0         1       2
```

8. Enter the **show ip ospf summary all-vrfs total** command to display cumulative summary output for OSPFv2 sessions.

```
device# show ip ospf summary all-vrfs total
-------------------------------------------
        IPv4 OSPF VRFs Summary Total
-------------------------------------------
Number of VRFs: 1
Number of Interfaces: 200
Number of Neighbors: 200
Number of Neighbors in Full state: 200
Number of LSAs: 182600
Number of Routes: 102600
```

9. Enter the **show ip ospf traffic** command to display OSPF traffic details.

```
device# show ip ospf traffic

                Packets Received         Packets Sent
Hello                       3943                 3936
Database                       6                    6
LSA Req                        1                    2
LSA Upd                       18                   20
LSA Ack                       17                   16
No Packet Errors!
```

Supported scale for OSPFv2 and performance considerations

Scalability of OSPFv2 gets limited by availability of resources like Memory, CPU and hardware tables. While increasing the scale, you need to consider time taken for adjacency formation and route convergence.

The supported scale for OSPFv2 across all VRF instances is as follows:

TABLE 40 Supported scale across all VRFs

Description	Number supported
Maximum number of OSPFv2 VRF Instances	1024
Maximum number of OSPFv2 Areas	1024
Number of OSPFv2 Interfaces	2048
Number of OSPFv2 Neighbors	2048
Maximum Number of OSPFv2 Routes	100k (102400)

The supported scale for OSPFv2 per VRF instance is as follows:

TABLE 41 Supported scale per VRF

Description	Number supported
Maximum number of OSPFv2 Areas	200
Number of OSPFv2 Interfaces	200
Number of OSPFv2 Neighbors	200
Maximum Number of OSPFv2 Routes	100k (102400)

Displaying OSPFv2 results

The **show ip ospf** command and its variations can be used to display information about OSPFv2 configurations.

Use one or more of the following commands to verify OSPFv2 information. Using the **show ip ospf** command is optional, and the variations of the command can be entered in any order.

1. In privileged EXEC mode, enter the **show ip ospf** command to display general OSPFv2 information.

   ```
   device# show ip ospf
   ```

   The example output displays general OSPFv2 information and indicates that the device is not operating as an ASBR. If the device is not operating as an ASBR, there is no information about redistribution in the output.
2. Enter the **show ip ospf area** command to display information about an OSPFv2 area.

   ```
   device# show ip ospf area
   ```

   The example output displays detailed output for OSPFv2 Area 0.
3. Enter the **show ip ospf interface** command to display OSPFv2 interface information.

   ```
   device# show ip ospf interface
   ```

   The example output displays detailed output for OSPFv2 interfaces.
4. Enter the **show ip ospf interface** command with the **brief** parameter to display summarized OSPFv2 interface information.

   ```
   device# show ip ospf interface brief
   ```

   The example output displays summarized output for OSPFv2 interfaces.
5. Enter the **show ip ospf neighbor** command to display OSPFv2 neighbor information for the device.

   ```
   device# show ip ospf neighbor
   ```

   The example output displays output for an OSPFv2 neighbor.
6. Enter the **show ip ospf routes** command to display information about OSPFv2 routes.

   ```
   device# show ip ospf routes
   ```

7. Enter the **show ip ospf summary** command to display summary output for OSPFv2 sessions.

   ```
   device# show ip ospf summary
   ```

8. Enter the **show ip ospf traffic** command to display OSPF traffic details.

   ```
   device# show ip ospf traffic
   ```

OSPFv3

- OSPFv3 overview 537
- OSPFv3 considerations and limitations 538
- Configuring the router ID 538
- Enabling OSPFv3 538
- Configuring OSPFv3 539
- OSPFv3 areas 539
- Virtual links 547
- OSPFv3 route redistribution 551
- Default route origination 553
- Disabling and re-enabling OSPFv3 event logging 554
- Disabling and re-enabling OSPFv3 event logging 555
- Filtering OSPFv3 routes 555
- SPF timers 559
- OSPFv3 administrative distance 560
- Changing the reference bandwidth for the cost on OSPFv3 interfaces 561
- OSPFv3 LSA refreshes 562
- External route summarization 563
- OSPFv3 over VRF 564
- Assigning OSPFv3 areas in a non-default VRF 567
- Setting all OSPFv3 interfaces to the passive state 568
- OSPFv3 graceful restart helper 569
- OSPFv3 non-stop routing 571
- OSPFv3 max-metric router LSA 572
- IPsec for OSPFv3 573
- Displaying OSPFv3 results 582
- Displaying OSPFv3 results 585
- Displaying OSPFv3 results 588
- Displaying OSPFv3 results 594
- Clearing OSPFv3 redistributed routes 596

OSPFv3 overview

Open Shortest Path First (OSPF) is a link-state routing protocol. Each OSPF device originates link-state advertisement (LSA) packets to describe its link information. These LSAs are flooded throughout the OSPF area. The flooding algorithm ensures that every device in the area has an identical database. Each device in the area then calculates a Shortest Path Tree (SPT) that shows the shortest distance to every other device in the area, using the topology information in the Link State database..

IPv6 supports OSPF Version 3 (OSPFv3), which functions similarly to OSPFv2, the version that IPv4 supports, except for the following enhancements:

- Support for IPv6 addresses and prefixes.
- Ability to configure several IPv6 addresses on a device interface. (While OSPFv2 runs per IP subnet, OSPFv3 runs per link. In general, you can configure several IPv6 addresses on a router interface, but OSPFv3 forms one adjacency per interface only, using the link local address of the interface as the source for OSPF protocol packets. On virtual links, OSPFv3 uses the global IP address as the source. OSPFv3 imports all or none of the address prefixes configured on a router interface. You cannot select the addresses to import.)
- Ability to run one instance of OSPFv2 and one instance of OSPFv3 concurrently on a link.

- Support for IPv6 link-state advertisements (LSAs).

  NOTE
  Although OSPFv2 and OSPFv3 function in a similar manner, Extreme has implemented the user interface for each version independently of the other. Therefore, any configuration of OSPFv2 features will not affect the configuration of OSPFv3 features and vice versa.

OSPFv3 considerations and limitations

There are a number of things to consider when configuring OSPFv3.

- OSPFv3 must be configured in a VCS environment.
- OSPFv3 can be configured on either a point-to-point or broadcast network.
- OSPFv3 can be enabled on the following interfaces: gigabitethernet, tengigabitethernet, fortygigabitethernet, hundredgigabitethenet, loopback, and ve.
- On enabling OSPFv3 over a loopback interface, the network is advertised as a stub network in the router LSA for the attached area. OSPFv3 control packets, such as *hellos* , are not transmitted on loopback interfaces and adjacencies will not form.

Configuring the router ID

When configuring OSPFv3, the router ID for a device must be specified.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config) ip router-id 10.11.12.13
   device(config-rbridge-id-122)# ip router-id 10.11.12.13
   ```

The following example configures the router ID for a device.

```
device# configure terminal
device(config)# ip router-id 10.11.12.13

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.11.12.13
```

Enabling OSPFv3

When OSPFv3 is enabled on a device, the device enters OSPFv3 router configuration mode. Several commands can then be accessed that allow the configuration of OSPFv3.

Before enabling the device to run OSPFv3, you must perform the following steps:

- Enable the forwarding of IPv6 traffic on the device using the **ipv6 unicast-routing** command.

- Enable IPv6 on each interface on which you plan to enable OSPFv3. You enable IPv6 on an interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip router-id** command to specify the router ID.

```
device(config) ip router-id 10.11.12.13
```

```
device(config-rbridge-id-122)# ip router-id 10.11.12.13
```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

```
device(config-rbridge-id-122)# ipv6 router ospf
```

The following example enables OSPFv3 on a device.

```
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 router ospf
device(config-ospf6-router)#

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.11.12.13
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)#

device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)#
```

Configuring OSPFv3

A number of steps are required when configuring OSPFv3:

- Configure the router ID.
- Enable OSPFv3 globally.
- Assign OSPFv3 areas.
- Assign OSPFv3 areas to interfaces.

OSPFv3 areas

After OSPFv3 is enabled, you can assign OSPFv3 areas. You can specify the area id in plain number format , such as "area 1", or in ipv4 address format, such as 10.1.1.1. Each device interface can support one area.

You can configure up to 200 OSPFv3 areas.

NOTE
You can assign only one area on a device interface.

NOTE
You are required to configure a router ID when running only IPv6 routing protocols.

NOTE
By default, the router ID is the IPv4 address configured on the lowest-numbered loopback interface. If the device does not have a loopback interface, the default router ID is the highest-numbered IPv4 address configured on the device. You can also configure router id using the **ip router-id** command.

Backbone area

The backbone area (also known as area 0 or area 0.0.0.0) forms the core of OSPF networks. All other areas should be connected to the backbone area either by a direct link or by virtual link configuration. Routers that have interfaces in both backbone area and (at least one) non-backbone area are called Area Border Routers (ABR). Inter area routing happens via ABRs.

The backbone area is the logical and physical structure for the OSPF domain and is attached to all non-zero areas in the OSPF domain.

The backbone area is responsible for distributing routing information between non-backbone areas. The backbone must be contiguous, but it does not need to be physically contiguous; backbone connectivity can be established and maintained through the configuration of virtual links.

Area range

You can further consolidate routes at an area boundary by defining an area range. The area range allows you to assign an aggregate address to a range of IP and IPv6 addresses.

This aggregate value becomes the address that is advertised instead of all the individual addresses it represents being advertised. Only this aggregate or summary address is advertised into other areas instead of all the individual addresses that fall in the configured range. Area range configuration can considerably reduce the number of Type 3 summary LSAs advertised by a device. You have the option of adding the cost to the summarized route. If you do not specify a value, the cost value is the default range metric calculation for the generated summary LSA cost. You can temporarily pause route summarization from the area by suppressing the type 3 LSA so that the component networks remain hidden from other networks.

You can assign up to 32 ranges in an OSPF area.

You can assign up to 4 ranges in an OSPFv3 area.

Area types

OSPFv3 areas can be normal, a stub area, a totally stubby area (TSA), or a not-so-stubby area (NSSA).

- Normal: OSPFv3 devices within a normal area can send and receive external link-state advertisements (LSAs).
- Stub: OSPFv3 devices within a stub area cannot send or receive External LSAs. In addition, OSPF devices in a stub area must use a default route to the area's Area Border Router (ABR) to send traffic out of the area.
- TSA: A form of stub area, where Type 3 summary routes are also not propagated in addition to Type 5 external routes.

- NSSA: A form of stub area, where Type 5 external routes by Autonomous System Boundary Routers (ASBRs) outside this area are not propagated, but where it is allowed to have an ASBR in the area, that can advertise external information.
  - ASBRs redistribute (import) external routes into the NSSA as type 7 LSAs. Type 7 External LSAs are a special type of LSA generated only by ASBRs within an NSSA, and are flooded to all the routers within only that NSSA.
  - One of the ABRs of the NSSA area is selected as a NSSA translator, and this router translates the area-specific Type 7 LSAs to Type 5 external LSAs which can be flooded throughout the Autonomous System (except NSSA and stub areas).

When an NSSA contains more than one ABR, OSPFv3 elects one of the ABRs to perform the LSA translation for NSSA. OSPF elects the ABR with the highest router ID. If the elected ABR becomes unavailable, OSPFv3 automatically elects the ABR with the next highest router ID to take over translation of LSAs for the NSSA. The election process for NSSA ABRs is automatic.

Assigning OSPFv3 areas

Areas can be assigned as OSPFv3 areas.

Enable IPv6 on each interface on which you plan to enable OSPFv3. You enable IPv6 on an interface by configuring an IP address or explicitly enabling IPv6 on that interface.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config) ip router-id 10.11.12.13
   device(config-rbridge-id-122)# ip router-id 10.11.12.13
   ```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config-rbridge-id-122)# ipv6 router ospf
   device(config)# ipv6 router ospf
   ```

5. Enter the **area** command to define an OSPFv3 area ID.

   ```
   device(config-ospf6-router)# area 0
   device(config-ipv6-router-ospf-vrf-default-vrf)# area 0
   ```

6. Enter the **area** command to define a second OSPFv3 area ID.

   ```
   device(config-ospf6-router)# area 10.1.1.1
   device(config-ipv6-router-ospf-vrf-default-vrf)# area 10.1.1.1
   ```

The following example assigns an OSPFv3 ID to two areas. One of the areas is assigned by decimal number. The second area is assigned by IP address.

```
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 router ospf
device(config-ospf6-router)# area 0
device(config-ospf6-router)# area 10.1.1.1

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.11.12.13
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 0
device(config-ipv6-router-ospf-vrf-default-vrf)# area 10.1.1.1

device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 0
device(config-ipv6-router-ospf-vrf-default-vrf)# area 10.1.1.1
```

Assigning OSPFv3 areas to interfaces

Defined OSPFv3 areas can be assigned to device interfaces.

Ensure that OSPFv3 areas are assigned.

> **NOTE**
> All device interfaces must be assigned to one of the defined areas on an OSPFv3 device. When an interface is assigned to an area, all corresponding subnets on that interface are automatically included in the assignment.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface** command and specify an interface.

   ```
   device(config)# interface ethernet 1/1
   device(config)# interface ethernet 0/1
   device(config)# interface tengigabitethernet 1/0/1
   ```

3. Enter the **ipv6 address** command to add an IPv6 address to the interface.

   ```
   device(config-if-e1000/1/1)# ipv6 address 2001:1:0::1:1/64
   device(config-if-te-1/0/1)# ipv6 address 2001:1:0::1:1/64
   device(conf-if-eth-1/1)# ipv6 address 2001:1:0::1:1/64
   device(conf-if-eth-0/1)# ipv6 address 2001:1:0::1:1/64
   ```

4. Enter the **ipv6 ospf area** command.

   ```
   device(config-if-e1000/1/1)# ipv6 ospf area 0
   device(config-if-te-1/0/1)# ipv6 ospf area 0
   device(conf-if-eth-1/1)# ipv6 ospf area 0
   device(conf-if-eth-0/1)# ipv6 ospf area 0
   ```

   Area 0 is assigned to the specified interface with the IPv6 address of 2001:1:0:1::1/64.

5. Enter the **exit** command to return to global configuration mode.

```
device(config-if-e1000/1/1)# exit
device(config-if-te-1/0/1)# exit
device(conf-if-eth-1/1)# exit
device(conf-if-eth-0/1)# exit
```

6. Enter the **interface** command and specify an interface.

```
device(config)# interface ethernet 2/1
device(config)# interface ethernet 0/2
device(config)# interface tengigabitethernet 1/0/2
```

7. Enter the **ipv6 address** command to add an IPv6 address to the interface.

```
device(config-if-e1000/2/1)# ipv6 address 2001:1:0::2:1/64
device(config-if-te-1/0/2)# ipv6 address 2001:1:0::2:1/64
device(conf-if-eth-2/1)# ipv6 address 2001:1:0::2:1/64
device(conf-if-eth-0/2)# ipv6 address 2001:1:0::2:1/64
```

8. Enter the **ipv6 ospf area** command.

```
device(config-if-e1000/2/1)# ipv6 ospf area 1
device(config-if-te-1/0/2)# ipv6 ospf area 1
device(conf-if-eth-2/1)# ipv6 ospf area 1
device(conf-if-eth-0/2)# ipv6 ospf area 1
```

Area 1 is assigned to the specified interface with the IPv6 address of 2001:1:0:2::1/64.

The following example configures and enables OSPFv3 on two specified interfaces, and assigns an interface to two router areas.

```
device# configure terminal
device(config)# interface ethernet 1/1
device(config-if-e1000/1/1)# ipv6 address 2001:1:0::1:1/64
device(config-if-e1000/1/1)# ipv6 ospf area 0
device(config-if-e1000/1/1)# exit
device(config)# interface ethernet 2/1
device(config-if-e1000/2/1)# ipv6 address 2001:1:0::2:1/64
device(config-if-e1000/2/1)# ipv6 ospf area 1

device# configure terminal
device(config)# interface tengigabitethernet 1/0/1
device(config-if-te-1/0/1)# ipv6 address 2001:1:0::1:1/64
device(config-if-te-1/0/1)# ipv6 ospf area 0
device(config-if-te-1/0/1)# exit
device(config)# interface tengigabitethernet 1/0/2
device(config-if-te-1/0/2)# ipv6 address 2001:1:0::2:1/64
device(config-if-te-1/0/2)# ipv6 ospf area 1

device# configure terminal
device(config)# interface ethernet 1/1
device(conf-if-eth-1/1)# ipv6 address 2001:1:0::1:1/64
device(conf-if-eth-1/1)# ipv6 ospf area 0
device(conf-if-eth-1/1)# exit
device(config)# interface ethernet 2/1
device(conf-if-eth-2/1)# ipv6 address 2001:1:0::2:1/64
device(conf-if-eth-2/1)# ipv6 ospf area 1

device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# ipv6 address 2001:1:0::1:1/64
device(conf-if-eth-0/1)# ipv6 ospf area 0
device(conf-if-eth-0/1)# exit
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# ipv6 address 2001:1:0::2:1/64
device(conf-if-eth-0/2)# ipv6 ospf area 1
```

Stub area and totally stubby area

A stub area is an area in which advertisements of external routes are not allowed, reducing the size of the database. A totally stubby area (TSA) is a stub area in which summary link-state advertisement (type 3 LSAs) are not sent. A default summary LSA, with a prefix of 0.0.0.0/0 is originated into the stub area by an ABR, so that devices in the area can forward all traffic for which a specific route is not known, via ABR.

A stub area disables advertisements of external routes. By default, the ABR sends summary LSAs (type 3 LSAs) into stub areas. You can further reduce the number of LSAs sent into a stub area by configuring the device to stop sending type 3 LSAs into the area. You can disable the summary LSAs to create a TSA when you are configuring the stub area or after you have configured the area.

The ABR of a totally stubby area disables origination of summary LSAs into this area, but still accepts summary LSAs from OSPF neighbors and floods them to other neighbors.

When you enter the **area stub** command with the **no-summary** keyword and specify an area to disable the summary LSAs, the change takes effect immediately. If you apply the option to a previously configured area, the device flushes all the summary LSAs it has generated (as an ABR) from the area with the exception of the default summary LSA originated. This default LSA is needed for the internal routers, since external routes are not propagated to them.

> NOTE
> Stub areas and TSAs apply only when the device is configured as an Area Border Router (ABR) for the area. To completely prevent summary LSAs from being sent to the area, disable the summary LSAs on each OSPF router that is an ABR for the area.

Configuring a stub area

OSPFv3 areas can be defined as stub areas with modifiable parameters.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config) ip router-id 10.4.4.4
   device(config-rbridge-id-122)# ip router-id 10.4.4.4
   ```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config)# ipv6 router ospf
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

5. Enter the **area stub** command and specify a metric value.

   ```
   device(config-ospf6-router)# area 4 stub 100
   device(config-ipv6-router-ospf-vrf-default-vrf)# area 4 stub 100
   ```

   Area 4 is defined as a stub area with an additional cost of 100.

The following example sets an additional cost of 100 on a stub area defined as 4.

```
device# configure terminal
device(config)# ip router-id 10.4.4.4
device(config)# ipv6 router ospf
device(config-ospf6-router)# area 4 stub 100

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.4.4.4
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 4 stub 100

device# configure terminal
device(config)# ip router-id 10.4.4.4
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 4 stub 100
```

Not-so-stubby area

A not-so-stubby-area (NSSA) is an OSPFv3 area that provides the benefits of stub areas with the extra capability of importing external route information. OSPFv3 does not flood external routes from other areas into an NSSA, but does translate and flood route information from the NSSA into other areas such as the backbone.

NSSAs are especially useful when you want to aggregate type 5 External LSAs (external routes) before forwarding them into an OSPFv3 area. When you configure an NSSA, you can specify an address range for aggregating the external routes that the ABR of the NSSAs exports into other areas.

The OSPFv3 specification (RFC 2328) prohibits the advertising of type 5 LSAs and requires OSPFv3 to flood type 5 LSAs throughout a routing domain.

If the router is an ABR, you can prevent any type 3 and type 4 LSA from being injected into the area by configuring a nssa with the **no-summary** parameter. The only exception is that a default route is injected into the NSSA by the ABR, and strictly as a type 3 LSA. The default type 7 LSA is not originated in this case.

By default, the device's NSSA translator role is set to candidate and the router participates in NSSA translation election, if it is an ABR. You can also configure the NSSA translator role.

In the case where an NSSA ABR is also an ASBR, the default behavior is that it originates type 5 LSAs into normal areas and type 7 LSAs into an NSSA. But you can prevent an NSSA ABR from generating type 7 LSAs into an NSSA by configuring the **no-redistribution** parameter.

Configuring an NSSA

OSPFv3 areas can be defined as NSSA areas with configurable parameters.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config) ip router-id 10.3.3.3
   device(config-rbridge-id-122)# ip router-id 10.3.3.3
   ```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config)# ipv6 router ospf
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

5. Enter the **area nssa** command with the **default-information-originate** keyword and specify a cost.

   ```
   device(config-ospf6-router)# area 3 nssa default-information-originate metric 33
   device(config-ipv6-router-ospf-vrf-default-vrf)# area 3 nssa default-information-originate metric 33
   ```

   Area 3 is defined as an NSSA with the default route option and an additional cost of 33.

The following example sets an additional cost of 33 on an NSSA defined as 3.

```
device# configure terminal
device(config)# ip router-id 10.3.3.3
device(config)# ipv6 router ospf
device(config-ospf6-router)# area 3 nssa default-information-originate metric 33

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.3.3.3
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 3 nssa default-information-originate metric 33

device# configure terminal
device(config)# ip router-id 10.3.3.3
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 3 nssa default-information-originate metric 33
```

LSA types for OSPFv3

Communication among OSPFv3 areas is provided by means of link-state advertisements (LSAs). OSPFv3 supports a number of types of LSAs:

- Router LSAs (Type 1)
- Network LSAs (Type 2)
- Interarea-prefix LSAs for ABRs (Type 3)
- Interarea-router LSAs for ASBRs (Type 4)
- Autonomous system External LSAs (Type 5)
- NSSA External LSAs (Type 7)
- Link LSAs (Type 8)
- Intra-area-prefix LSAs (Type 9)

For more information about these LSAs, refer to RFC 5340.

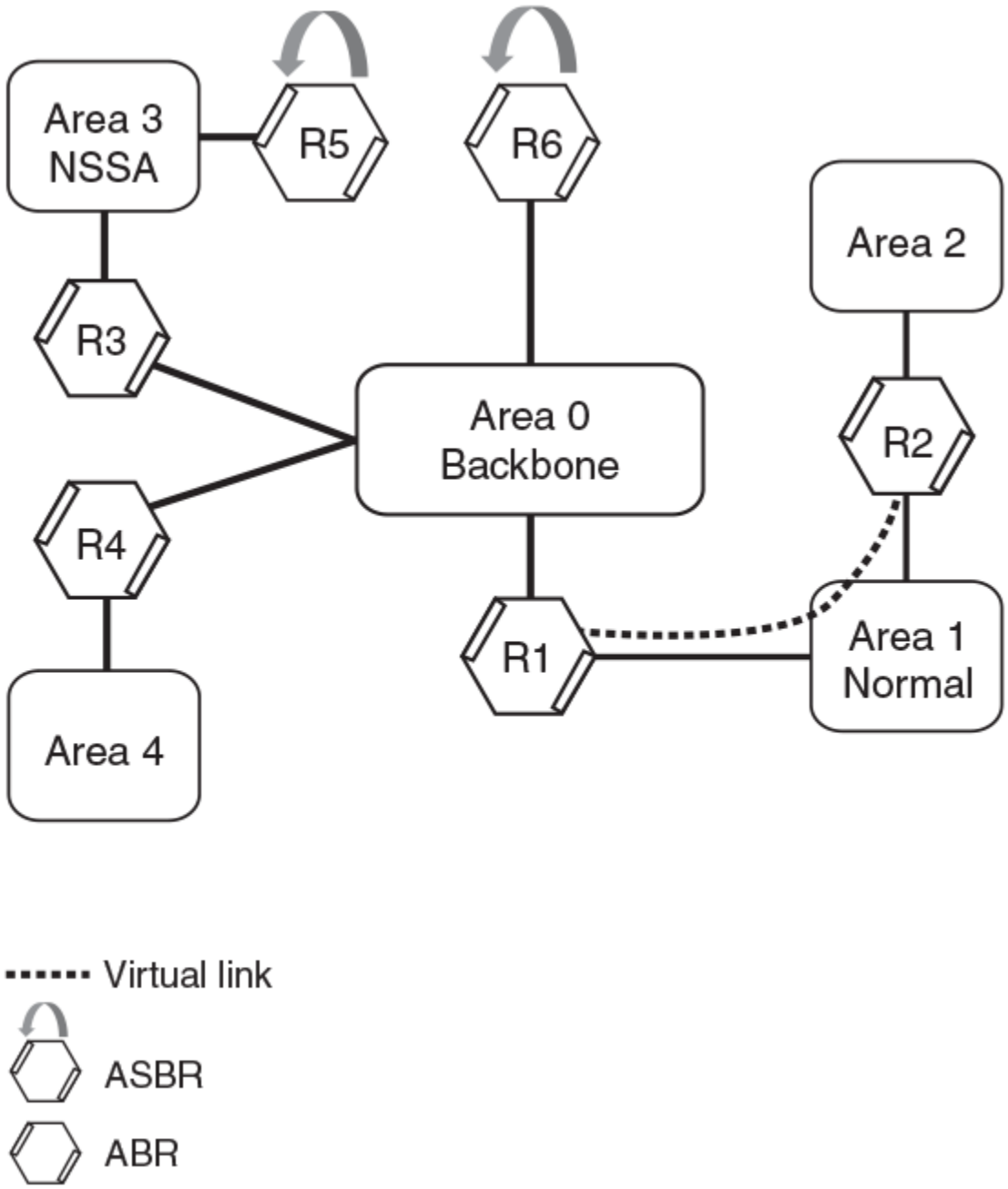
Virtual links

All ABRs must have either a direct or indirect link to an OSPFv3 backbone area (0 or 0.0.0.0). If an ABR does not have a physical link to a backbone area, you can configure a virtual link from the ABR to another router within the same area that has a physical connection to the backbone area.

The path for a virtual link is through an area shared by the neighbor ABR (router with a physical backbone connection) and the ABR requiring a logical connection to the backbone.

In the following figure, a virtual link has been created between ABR1 and ABR2. ABR1 has a direct link to the backbone area, while ABR2 has an indirect link to the backbone area through Area 1.

FIGURE 77 OSPFv3 virtual link



Two parameters must be defined for all virtual links—transit area ID and neighbor router:

- The transit area ID represents the shared area of the two ABRs and serves as the connection point between the two routers. This number should match the area ID value.
- The neighbor router is the router ID of the device that is physically connected to the backbone when assigned from the router interface requiring a logical connection. The neighbor router is the router ID (IPv4 address) of the router requiring a logical connection to the backbone when assigned from the router interface with the physical connection.

When you establish an area virtual link, you must configure it on both ends of the virtual link. For example, imagine that ABR1 in Area 1 and Area 2 is cut off from the backbone area (Area 0). To provide backbone access to ABR1, you can add a virtual link between ABR1 and ABR2 in Area 1 using Area 1 as a transit area. To configure the virtual link, you define the link on the router that is at each end of the link. No configuration for the virtual link is required on the routers in the transit area.

Virtual links cannot be configured in stub areas and NSSAs.

Virtual link source address assignment

When devices at both ends of a virtual link communicate with one another, a global IPv6 address is automatically selected for each end device and this address is advertised into the transit area as an intra-area-prefix LSA.

The automatically selected global IPv6 address for that router is the first global address of any loopback interface in that transit area. If no global IPv6 address is available on a loopback interface in the area, the first global IPv6 address of the lowest-numbered interface in the UP state (belonging to the transit area) is assigned. If no global IPv6 address is configured on any of the OSPFv3 interfaces in the transit area, the virtual links in the transit area do not operate. The automatically selected IPv6 global address is updated whenever the previously selected IPv6 address of the interface changes, is removed, or if the interface goes down.

> NOTE
> The existing selected virtual link address does not change because the global IPv6 address is now available on a loopback interface or a lower-numbered interface in the transit area. To force the global IPv6 address for the virtual link to be the global IPv6 address of a newly configured loopback, or a lower-numbered interface in the area, you must either disable the existing selected interface or remove the currently selected global IPv6 address from the interface.

Configuring virtual links

If an Area Border Router (ABR) does not have a physical link to a backbone area, a virtual link can be configured between that ABR and another device within the same area that has a physical link to a backbone area.

A virtual link is configured, and a virtual link endpoint on two devices, ABR1 and ABR2, is defined.

1. On ABR1, enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config) ip router-id 10.1.1.1
   device(config-rbridge-id-122)# ip router-id 10.1.1.1
   ```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config)# ipv6 router ospf
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

5. Enter the **area** command to assign an OSPFv3 area ID.

   ```
   device(config-ospf6-router)# area 0
   device(config-ipv6-router-ospf-vrf-default-vrf)# area 0
   ```

6. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ospf6-router)# area 1
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1
```

7. Enter the **area virtual-link** command and the ID of the OSPFv3 device at the remote end of the virtual link to configure the virtual link endpoint.

```
device(config-ospf6-router)# area 1 virtual-link 10.2.2.2
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.2.2.2
```

8. On ABR2, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

9. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 104
```

10. Enter the **ip router-id** command to specify the router ID.

```
device(config) ip router-id 10.2.2.2
device(config-rbridge-id-104)# ip router-id 10.2.2.2
```

11. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
device(config-rbridge-id-104)# ipv6 router ospf
```

12. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ospf6-router)# area 1
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1
```

13. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ospf6-router)# area 2
device(config-ipv6-router-ospf-vrf-default-vrf)# area 2
```

14. Enter the **area virtual-link** command and the ID of the OSPFv3 device at the remote end of the virtual link to configure the virtual link endpoint.

```
device(config-ospf6-router)# area 1 virtual-link 10.1.1.1
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1
```

The following example configures a virtual link between two devices.

```
ABR1:
device1# configure terminal
device1(config)# ip router-id 10.1.1.1
device1(config)# ipv6 router ospf
device1(config-ospf6-router)# area 0
device1(config-ospf6-router)# area 1
device1(config-ospf6-router)# area 1 virtual-link 10.2.2.2

ABR2:
device2# configure terminal
device2(config)# ip router-id 10.2.2.2
device2(config)# ipv6 router ospf
device2(config-ospf6-router)# area 1
device2(config-ospf6-router)# area 2
device2(config-ospf6-router)# area 1 virtual-link 10.1.1.1

ABR1:
device1# configure terminal
device1(config)# rbridge-id 122
device1(config-rbridge-id-122)# ip router-id 10.1.1.1
device1(config-rbridge-id-122)# ipv6 router ospf
device1(config-ipv6-router-ospf-vrf-default-vrf)# area 0
device1(config-ipv6-router-ospf-vrf-default-vrf)# area 1
device1(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.2.2.2

ABR2:
device2# configure terminal
device2(config)# rbridge-id 104
device2(config-rbridge-id-104)# ip router-id 10.2.2.2
device2(config-rbridge-id-104)# ipv6 router ospf
device2(config-ipv6-router-ospf-vrf-default-vrf)# area 1
device2(config-ipv6-router-ospf-vrf-default-vrf)# area 2
device2(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1

ABR1:
device1# configure terminal
device1(config)# ip router-id 10.1.1.1
device1(config)# ipv6 router ospf
device1(config-ipv6-router-ospf-vrf-default-vrf)# area 0
device1(config-ipv6-router-ospf-vrf-default-vrf)# area 1
device1(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.2.2.2

ABR2:
device2# configure terminal
device2(config)# ip router-id 10.2.2.2
device2(config)# ipv6 router ospf
device2(config-ipv6-router-ospf-vrf-default-vrf)# area 1
device2(config-ipv6-router-ospf-vrf-default-vrf)# area 2
device2(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1
```

OSPFv3 route redistribution

Routes from various sources can be redistributed into OSPFv3. These routes can be redistributed in a number of ways.

You can configure the device to redistribute routes from the following sources into OSPFv3:

- IPv6 static routes
- Directly connected IPv6 networks
- BGP4+
- IPv6 IS-IS
- RIPng

You can redistribute routes in the following ways:

- By route types. For example, the device redistributes all IPv6 static routes.
- By using a route map to filter which routes to redistribute. For example, the device redistributes specified IPv6 static routes only.

NOTE
You must configure the route map before you configure a redistribution filter that uses the route map.

NOTE
For an external route that is redistributed into OSPFv3 through a route map, the metric value of the route remains the same unless the metric is set by the **set metric** command inside the route map or the **default-metric** command. For a route redistributed without using a route map, the metric is set by the metric parameter if set or the **default-metric** command if the metric parameter is not set.

NOTE
The CLI VRF-lite-capability present in OSPFv2 is not present in OSPFv3. Therefore, when a BGP route is redistributed from an MPLS domain into OSPFv3 and the DN (down) bit is set, the routes must be installed in the OSPFv3 routing table. Such routes could get propagated back into the MLPS cloud if there are OSPFv3 back-door links configured.

Redistributing routes into OSPFv3

OSPFv3 routes can be redistributed, and the routes to be redistributed can be specified.

The redistribution of both static routes and BGP routes into OSPFv3 is configured on device1. The redistribution of connected routes into OSPFv3 is configured on device2, and the connected routes to be redistributed are specified.

1. On device1, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
device(config-rbridge-id-122)# ipv6 router ospf
```

4. Enter the **redistribute** command with the **static** parameter to redistribute static routes.

```
device(config-ospf6-router)# redistribute static
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute static
```

5. Enter the **redistribute** command with the **bgp** parameter to redistribute static routes.

```
device(config-ospf6-router)# redistribute bgp
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute bgp
```

6. On device2, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

7. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

8. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf

device(config-rbridge-id-122)# ipv6 router ospf
```

9. Enter the **redistribute** command with the **connected** and **route-map** parameters to redistribute connected routes and specify a route map.

```
device(config-ospf6-router)# redistribute connected route-map rmap1

device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute connected route-map rmap1
```

The following example redistributes static and BGP routes routes into OSPFv3 on a device.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# redistribute static
device(config-ospf6-router)# redistribute bgp

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute static
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute bgp

device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute static
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute bgp
```

The following example redistributes connected routes into OSPFv3 on a device and specifies a route map.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# redistribute connected route-map rmap1

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute connected route-map rmap1

device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute connected route-map rmap1
```

Default route origination

When the device is an OSPFv3 Autonomous System Boundary Router (ASBR), you can configure it to automatically generate a default external route into an OSPFv3 routing domain.

By default, a device does not advertise the default route into the OSPFv3 domain. If you want the device to advertise the OSPFv3 default route, you must explicitly enable default route origination. When you enable OSPFv3 default route origination, the device advertises a type 5 default route that is flooded throughout the autonomous system, with the exception of stub areas.

The device advertises the default route into OSPFv3 even if OSPFv3 route redistribution is not enabled, and even if the default route is learned through an IBGP neighbor. The device does not, however, originate the default route if the active default route is learned from an OSPFv3 router in the same domain.

NOTE
The device does not advertise the OSPFv3 default route, regardless of other configuration parameters, unless you explicitly enable default route origination.

If default route origination is enabled and you disable it, the default route originated by the device is flushed. Default routes generated by other OSPFv3 devices are not affected. If you re-enable the default route origination, the change takes effect immediately and you do not need to reload the software.

Configuring default external routes

OSPFv3 default routes can be created and advertised.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config)# ipv6 router ospf

device(config-rbridge-id-122)# ipv6 router ospf
```

4. Enter the **default-information-originate** command with the **always**, **metric**, and **metric-type** parameters.

```
device(config-ospf6-router)# default-information-originate always metric 2 metric-type type1

device(config-ipv6-router-ospf-vrf-default-vrf)# default-information-originate always metric 2
metric-type type1
```

A default type 1 external route with a metric of 2 is created and advertised.

The following example creates and advertises a default route with a metric of 2 and a type 1 external route.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# default-information-originate always metric 2 metric-type type1

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# default-information-originate always metric 2 metric-type
type1

device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# default-information-originate always metric 2 metric-type
type1
```

Disabling and re-enabling OSPFv3 event logging

OSPFv3 event logging, such as neighbor state changes and database overflow conditions, can be disabled and re-enabled.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config)# ipv6 router ospf
```

3. Enter the **no log-status-change** command to disable the logging of OSPFv3 events.

```
device(config-ospf6-router)# no log-status-change
```

The following example re-enables the logging of OSPFv3 events.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# log-status-change
```

Disabling and re-enabling OSPFv3 event logging

OSPFv3 event logging, such as neighbor state changes and database overflow conditions, can be disabled and re-enabled.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 configuration mode and enable OSPFv3 globally.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

```
device(config)# ipv6 router ospf
```

4. Enter the **no log** command to disable the logging of OSPFv3 events.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# no log
```

The following example re-enables the logging of OSPFv3 events.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# log all
```

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# log all
```

Filtering OSPFv3 routes

You can filter the routes to be placed in the OSPFv3 route table by configuring distribution lists.

The functionality of OSPFv3 distribution lists is similar to that of OSPFv2 distribution lists. Refer to the **OSPFv2** chapter for more information.

The functionality of OSPFv3 distribution lists is similar to that of OSPFv2 distribution lists. However, unlike OSPFv2 distribution lists, which filter routes based on criteria specified in an Access Control List (ACL), OSPFv3 distribution lists can filter routes using information specified in an IPv6 prefix list or a route map.

Configuring an OSPFv3 distribution list using an IPv6 prefix list as input

An IPv6 prefix list can be used to filter OSPFv3 routes.

1. Enter the **show ipv6 ospf route** command to verify the OSPFv3 routes.

```
device> show ipv6 ospf route
  Current Route count: 5
    Intra: 3 Inter: 0 External: 2 (Type1 0/Type2 2)
    Equal-cost multi-path: 0
    Destination                    Options   Area             Cost Type2 Cost
    Next Hop Router                Outgoing Interface
*IA 2001:db8:1::/64                --------- 10.0.0.1           0 0
    ::                             ve 10
*E2 2001:db8:2::/64                --------- 0.0.0.0            10 0
    fe80::2e0:52ff:fe00:10         ve 10
*IA 2001:db8:3::/64                V6E---R-- 0.0.0.0            11 0
    fe80::2e0:52ff:fe00:10         ve 10
*IA 2001:db8:4::/64                --------- 0.0.0.0            10 0
    ::                             ve 11
*E2 2001:db8:5::/64                --------- 0.0.0.0            10 0
    fe80::2e0:52ff:fe00:10         ve 10
```

2. Enter the **enable** command to access privileged EXEC mode.

```
device> enable
```

3. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

4. Enter the **ip router-id** command to specify the router ID.

```
device(config)# ip router-id 10.11.12.13
```

5. Enter the **ipv6 prefix-list** command, using the **deny** keyword and specify a name.

```
device(config)# ipv6 prefix-list filterOspfRoutes seq 5 deny 2001:db8:2::/642
```

An IPv6 prefix list called “filterOspfRoutes” that denies route 2001:db8:2::/64 is configured.

6. Enter the **ipv6 prefix-list** command using the **deny** keyword and specify a name. Use the **ge** keyword to specify a prefix length greater than or equal to the *ipv6-prefix/prefix-length* arguments. Use the **le** keyword to specify a prefix length less than or equal to the *ipv6-prefix/prefix-length* arguments.

```
device(config)# ipv6 prefix-list filterOspfRoutes seq 7 permit ::/0 ge 1 le 128
```

7. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

8. Enter the **distribute-list prefix-list** command, using the **in** keyword and specifying a name to configure a distribution list that applies the filterOspfRoutes prefix list globally.

```
device(config-ospf6-router)# distribute-list prefix-list filterOspfRoutes in
```

9. Enter the **exit** command until you return to user EXEC mode.

```
device(config-ospf6-router)# exit
```

10. Enter the **show ipv6 ospf route** command to verify that route 2001:db8:2::/64 is now omitted from the route table.

```
device> show ipv6 ospf route
  Current Route count: 4
    Intra: 3 Inter: 0 External: 1 (Type1 0/Type2 1)
    Equal-cost multi-path: 0
    Destination                    Options   Area             Cost Type2 Cost
    Next Hop Router                Outgoing Interface
*IA 2001:db8:1::/64                --------- 10.0.0.1            0 0
    ::                             ve 10
*IA 2001:db8:3::/64                V6E---R-- 0.0.0.0            11 0
    fe80::2e0:52ff:fe00:10         ve 10
*IA 2001:db8:4::/64                --------- 0.0.0.0            10 0
    ::                             ve 11
*E2 2001:db8:5::/64                --------- 0.0.0.0            10 0
    fe80::2e0:52ff:fe00:10         ve 10
```

The following example configures an IPv6 prefix list that is used to filter OSPFv3 routes. A distribution list is then configured and route 2001:db8:2::/64 is omitted from the OSPFv3 route table.

```
device> show ipv6 ospf route
device> enable
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 prefix-list filterOspfRoutes seq 5 deny 2001:db8:2::/642
device(config)# ipv6 prefix-list filterOspfRoutes seq 7 permit ::/0 ge 1 le 128
device(config)# ipv6 router ospf
device(config-ospf6-router)# distribute-list prefix-list filterOspfRoutes in
device(config-ospf6-router)# exit
device> show ipv6 ospf route
```

Configuring an OSPFv3 distribution list using a route map as input

A route map that matches internal routes can be used to filter OSPFv3 routes.

1. Enter the **show ipv6 ospf route** command to verify the OSPFv3 routes.

```
device# show ipv6 ospf route
  Current Route count: 5
    Intra: 3 Inter: 0 External: 2 (Type1 0/Type2 2)
    Equal-cost multi-path: 0
    Destination                    Options   Area             Cost Type2 Cost
    Next Hop Router                Outgoing Interface
*IA 2001:db8:1::/64                --------- 10.0.0.1            0 0
    ::                             ve 10
*E2 2001:db8:2::/64                --------- 0.0.0.0            10 0
    fe80::2e0:52ff:fe00:10         ve 10
*IA 2001:db8:3::/64                V6E---R-- 0.0.0.0            11 0
    fe80::2e0:52ff:fe00:10         ve 10
*IA 2001:db8:4::/64                --------- 0.0.0.0            10 0
    ::                             ve 11
*E2 2001:db8:5::/64                --------- 0.0.0.0            10 0
    fe80::2e0:52ff:fe00:10         ve 10
```

2. Enter the **enable** command to access privileged EXEC mode.

```
device> enable
```

3. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

4. Enter the **ip router-id** command to specify the router ID.

```
device(config)# ip router-id 10.11.12.13
```

5. Enter the **route-map** command, using the **permit** keyword and specify a name.

```
device(config)# route-map allowInternalRoutes permit 10
```

   A route map called “allowInternalRoutes” that permits a matching pattern is configured.

6. Enter the **match route-type** command, using the **internal** keyword to match internal route types in the route map instance.

```
device(config-routemap allowInternalRoutes)# match route-type internal
```

7. Enter the **exit** command to return to global configuration mode.

```
device(config-routemap allowInternalRoutes)# exit
```

8. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

9. Enter the **distribute-list route-map** command, using the **in** keyword and specifying a name to create a distribution list a distribution list using the configured route map “allowinternalroutes”.

```
device(config-ospf6-router)# distribute-list route-map allowinternalroutes in
```

10. Enter the **exit** command until you return to user EXEC mode.

```
device(config-ospf6-router)# exit
```

11. Enter the **show ipv6 ospf route** command to verify that the external routes are omitted from the OSPFv3 route table.

```
device> show ipv6 ospf route
  Current Route count: 3
    Intra: 3 Inter: 0 External: 0 (Type1 0/Type2 0)
    Equal-cost multi-path: 0
    Destination                    Options   Area             Cost Type2 Cost
    Next Hop Router                Outgoing Interface
*IA 2001:db8:3001::/64            --------- 10.0.0.1             0 0
    ::                             ve 10
*IA 2001:db8:3015::/64             V6E---R-- 0.0.0.0            11 0
    fe80::2e0:52ff:fe00:10         ve 10
*IA 2001:db8:3020::/64             --------- 0.0.0.0            10 0
    ::                             ve 11
```

The following example configures a route map that is used to filter OSPFv3 routes. A distribution list is then configured and external routes omitted from the OSPFv3 route table.

```
device> show ipv6 ospf route
device> enable
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# route-map allowInternalRoutes permit 10
device(config-routemap allowInternalRoutes)# match route-type internal
device(config-routemap allowInternalRoutes)# exit
device(config)# ipv6 router ospf
device(config-ospf6-router)# distribute-list route-map allowinternalroutes in
device(config-ospf6-router)# exit
device> show ipv6 ospf route
```

SPF timers

The device uses an SPF delay timer and an SPF hold-time timer to calculate the shortest path for OSPFv3 routes. The values for both timers can be changed.

The device uses the following timers when calculating the shortest path for OSPFv3 routes:

- SPF delay: When the device receives a topology change, it waits before starting a Shortest Path First (SPF) calculation. By default, the device waits 5 seconds. You can configure the SPF delay to a value from 0 through 65535 seconds. If you set the SPF delay to 0 seconds, the device immediately begins the SPF calculation after receiving a topology change.
- SPF delay: When the device receives a topology change, it waits before starting a Shortest Path First (SPF) calculation. By default, the device waits 0 seconds. You can configure the SPF delay to a value from 0 through 65535 seconds. When the SPF delay is set to 0 seconds, the device immediately begins the SPF calculation after receiving a topology change.
- SPF hold time: The device waits a specific amount of time between consecutive SPF calculations. By default, it waits 10 seconds. You can configure the SPF hold time to a value from 0 through 65535 seconds. If you set the SPF hold time to 0 seconds, the device does not wait between consecutive SPF calculations.
- SPF hold time: The device waits a specific amount of time between consecutive SPF calculations. By default, it waits 0 seconds. You can configure the SPF hold time to a value from 0 through 65535 seconds. When the SPF hold time is set to 0 seconds, the device does not wait between consecutive SPF calculations.

You can set the SPF delay and hold time to lower values to cause the device to change to alternate paths more quickly if a route fails. Note that lower values for these parameters require more CPU processing time.

You can change one or both of the timers.

> NOTE
> If you want to change only one of the timers, for example, the SPF delay timer, you must specify the new value for this timer as well as the current value of the SPF hold timer, which you want to retain. The device does not accept only one timer value.

Modifying SPF timers

The Shortest Path First (SPF) delay and hold time can be modified.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode..

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

4. Enter the **timers** command with the **spf** parameter.

   ```
   device(config-ospf6-router)# timers spf 1 5
   device(config-ipv6-router-ospf-vrf-default-vrf)# timers spf 1 5
   ```

   The SPF delay is changed to 1 second and the SPF hold time is changed to 5 seconds.

The following example changes the SPF delay and hold time.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# timers spf 1 5

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# timers spf 1 5

device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# timers spf 1 5
```

OSPFv3 administrative distance

Devices can learn about networks from various protocols and select a route based on the source of the route information. This decision can be influenced if the default administrative distance for OSPFv3 routes is changed. Consequently, the routes to a network may differ depending on the protocol from which the routes were learned.

You can influence the device's decision by changing the default administrative distance for OSPFv3 routes. You can configure a unique administrative distance for each type of OSPFv3 route. For example, you can configure the Extreme device to prefer a static route over an OSPFv3 inter-area route and to prefer OSPFv3 intra-area routes over static routes. The distance you specify influences the choice of routes when the device has multiple routes to the same network from different protocols. The device prefers the route with the lower administrative distance.

You can specify unique default administrative distances for the following OSPFv3 route types:

- Intra-area routes
- Inter-area routes
- External routes

NOTE
The choice of routes within OSPFv3 is not influenced. For example, an OSPFv3 intra-area route is always preferred over an OSPFv3 inter-area route, even if the intra-area route's distance is greater than the inter-area route's distance.

Configuring administrative distance based on route type

The default administrative distances for intra-area routes, inter-area routes, and external routes can be altered.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

4. Enter the **distance** command with the **intra-area** parameter.

```
device(config-ospf6-router)# distance intra-area 80
```

```
device(config-ipv6-router-ospf-vrf-default-vrf)# distance intra-area 80
```

   The administrative distance for intra-area routes is changed from the default to 80.

5. Enter the **distance** command with the **inter-area** parameter.

```
device(config-ospf6-router)# distance inter-area 90
```

```
device(config-ipv6-router-ospf-vrf-default-vrf)# distance inter-area 90
```

   The administrative distance for inter-area routes is changed from the default to 90.

6. Enter the **distance** command with the **external** parameter.

```
device(config-ospf6-router)# distance external 100
```

```
device(config-ipv6-router-ospf-vrf-default-vrf)# distance external 100
```

   The administrative distance for external routes is changed from the default to 100.

The following example changes the default administrative distances for intra-area routes, inter-area routes, and external routes.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# distance intra-area 80
device(config-ospf6-router)# distance inter-area 90
device(config-ospf6-router)# distance external 100

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# distance intra-area 80
device(config-ipv6-router-ospf-vrf-default-vrf)# distance inter-area 90
device(config-ipv6-router-ospf-vrf-default-vrf)# distance external 100

device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# distance intra-area 80
device(config-ipv6-router-ospf-vrf-default-vrf)# distance inter-area 90
device(config-ipv6-router-ospf-vrf-default-vrf)# distance external 100
```

Changing the reference bandwidth for the cost on OSPFv3 interfaces

The reference bandwidth for OSPFv3 can be altered, resulting in various costs.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

```
device(config)# ipv6 router ospf
```

4. Enter the **auto-cost reference-bandwidth** command to change the reference bandwidth.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# auto-cost reference-bandwidth 500

device(config-ospf6-router)# auto-cost reference-bandwidth 500
```

The following example changes the auto-cost reference bandwidth to 500.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# auto-cost reference-bandwidth 500

device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# auto-cost reference-bandwidth 500

device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# auto-cost reference-bandwidth 500
```

The reference bandwidth specified in this example results in the following costs:

- 10-Mbps port cost = 500/10 = 50
- 100-Mbps port cost = 500/100 = 5
- 1000-Mbps port cost = 500/1000 = 0.5, which is rounded up to 1
- 155-Mbps port cost = 500/155 = 3.23, which is rounded up to 4
- 622-Mbps port cost = 500/622 = 0.80, which is rounded up to 1
- 2488-Mbps port cost = 500/2488 = 0.20, which is rounded up to 1

The costs for 10-Mbps, 100-Mbps, and 155-Mbps ports change as a result of the changed reference bandwidth. Costs for higher-speed interfaces remain the same.

OSPFv3 LSA refreshes

To prevent a refresh from being performed each time an individual LSA's refresh timer expires, OSPFv3 LSA refreshes are delayed for a specified time interval. This pacing interval can be altered.

The device paces OSPFv3 LSA refreshes by delaying the refreshes for a specified time interval instead of performing a refresh each time an individual LSA's refresh timer expires. The accumulated LSAs constitute a group, which the device refreshes and sends out together in one or more packets.

The pacing interval, which is the interval at which the device refreshes an accumulated group of LSAs, is configurable in a range from 10 through 1800 seconds (30 minutes). The default is 240 seconds (4 minutes). Thus, every four minutes, the device refreshes the group of accumulated LSAs and sends the group together in the same packets.

The pacing interval is inversely proportional to the number of LSAs the device is refreshing and aging. For example, if you have approximately 10,000 LSAs, decreasing the pacing interval enhances performance. If you have a very small database (40 to 100 LSAs), increasing the pacing interval to 10 to 20 minutes may enhance performance only slightly.

Configuring the OSPFv3 LSA pacing interval

The interval between OSPFv3 LSA refreshes can be modified.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID t o enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

4. Enter the **timers** command with the **lsa-group-pacing** parameter.

   ```
   device(config-ospf6-router)# timers lsa-group-pacing 120
   device(config-ipv6-router-ospf-vrf-default-vrf)# timers lsa-group-pacing 120
   ```

   The OSPFv3 LSA pacing interval is changed to 120 seconds (two minutes).

The following example restores the pacing interval to the default value of 240 seconds (4 minutes).

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# no timers lsa-group-pacing

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# no timers lsa-group-pacing

device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# no timers lsa-group-pacing
```

External route summarization

An ASBR can be configured to advertise one external route as an aggregate for all redistributed routes that are covered by a specified IPv6 summary address range.

When you configure a summary address range, the range takes effect immediately. All the imported routes are summarized according to the configured summary address range. Imported routes that have already been advertised and that fall within the range are flushed out of the autonomous system and a single route corresponding to the range is advertised.

If a route that falls within a configured summary address range is imported by the device, no action is taken if the device has already advertised the aggregate route; otherwise, the device advertises the aggregate route. If an imported route that falls within a configured summary address range is removed by the device, no action is taken if there are other imported routes that fall within the same summary address range; otherwise, the aggregate route is flushed.

You can configure up to 32 summary address ranges.

You can configure up to 4 summary address ranges.

The device sets the forwarding address of the aggregate route to 0 and sets the tag to 0. If you delete a summary address range, the advertised aggregate route is flushed and all imported routes that fall within the range are advertised individually. If an external link-state

database (LSDB) overflow condition occurs, all aggregate routes and other external routes are flushed out of the autonomous system. When the device exits the external LSDB overflow condition, all the imported routes are summarized according to the configured address ranges.

NOTE
If you use redistribution filters in addition to summary address ranges, the device applies the redistribution filters to routes first, and then applies them to the summary address ranges.

NOTE
If you disable redistribution, all the aggregate routes are flushed, along with other imported routes.

NOTE
Only imported, type 5 external LSA routes are affected. A single type 5 LSA is generated and flooded throughout the autonomous system for multiple external routes.

OSPFv3 over VRF

OSPFv3 can run over multiple Virtual Routing and Forwarding (VRF) instances. OSPFv3 maintains multiple instances of the routing protocol to exchange route information among various VRF instances. A multi-VRF-capable router maps an input interface to a unique VRF, based on user configuration. These input interfaces can be physical or a virtual interface. By default, all input interfaces are attached to the default VRF instance. All OSPFv3 commands are available over default and nondefault VRF instances.

Multi-VRF for OSPF (also known as VRF-Lite for OSPF) provides a reliable mechanism for trusted VPNs to be built over a shared infrastructure. The ability to maintain multiple virtual routing or forwarding tables allows overlapping private IP addresses to be maintained across VPNs.

Enabling OSPFv3 in a non-default VRF

When OSPFv3 is enabled in a non-default VRF instance, the device enters OSPFv3 router VRF configuration mode. Several commands can then be accessed that allow the configuration of OSPFv3.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **vrf** command and specify a name to enter Virtual Routing and Forwarding (VRF) configuration mode and create a non-default VRF instance.

```
device(config)# vrf green
```

3. Enter the **rd** command, assigning an administrative number and arbitrary number the route, to distinguish a route for VRF green.

```
device(config-vrf-green)# rd 100:200
```

4. Enter the **ip router-id** command to specify the router ID.

```
device(config-vrf-green)# ip router-id 10.11.12.14
```

5. Enter the **address-family ipv6** command to enter IPv6 address-family configuration mode.

```
device(config-vrf-green)# address-family ipv6
```

6. Enter the **exit** command until you return to global configuration mode.

```
device(config-vrf-green-ipv6)# exit
```

7. Enter the **ipv6 router ospf** command and specify a VRF name to enter OSPFv3 router VRF configuration mode and enable OSPFv3 on a non-default VRF.

```
device(config)# ipv6 router ospf vrf green
```

The following example enables OSPFv3 in a non-default VRF.

```
device# configure terminal
device(config)# vrf green
device(config-vrf-green)# rd 100:200
device(config-vrf-green)# ip router-id 10.11.12.14
device(config-vrf-green)# address-family ipv6
device(config-vrf-green-ipv6)#
device(config-vrf-green-ipv6)# exit
device(config)# ipv6 router ospf vrf green
device(config-ospf6-router-vrf-green)#
```

Enabling OSPFv3 in a non-default VRF

When OSPFv3 is enabled in a non-default VRF instance, the device enters OSPFv3 router VRF configuration mode. Several commands can then be accessed that allow the configuration of OSPFv3.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **vrf** command and specify a name to enter Virtual Routing and Forwarding (VRF) configuration mode and create a non-default VRF instance.

```
device(config-rbridge-id-122)# vrf green
```

4. Enter the **ip router-id** command to specify the router ID.

```
device(config-vrf-green)# ip router-id 10.11.12.14
```

5. Enter the **address-family ipv6 unicast** command to enter IPv6 address-family configuration mode.

```
device(config-vrf-green)# address-family ipv6 unicast
```

6. Enter the **exit** command until you return to RBridge ID configuration mode.

```
device(vrf-ipv6-unicast)# exit
```

7. Enter the **ipv6 router ospf** command and specify a VRF name to enter OSPFv3 router VRF configuration mode and enable OSPFv3 on a non-default VRF.

```
device(config-rbridge-id-122)# ipv6 router ospf vrf green
```

The following example enables OSPFv3 in a non-default VRF.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# vrf green
device(config-vrf-green)# ip router-id 10.11.12.14
device(config-vrf-green)# address-family ipv6 unicast
device(vrf-ipv6-unicast)#
device(vrf-ipv6-unicast)# exit
device(config-vrf-green)# exit
device(config-rbridge-id-122)# ipv6 router ospf vrf green
device(config-ipv6-router-ospf-vrf-green)#
```

Enabling OSPFv3 in a non-default VRF

When OSPFv3 is enabled in a non-default VRF instance, the device enters OSPFv3 router VRF configuration mode. Several commands can then be accessed that allow the configuration of OSPFv3.

A non-default VRF instance has been configured.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 router ospf** command and specify a VRF name to enter OSPF router VRF configuration mode and enable OSPFv2 on a non-default VRF.

   ```
   device(config)# ipv6 router ospf vrf green
   ```

The following enables OSPFv3 in a non-default VRF.

```
device# configure terminal
device(config)# ipv6 router ospf vrf green
device(config-ipv6-router-ospf-vrf-green)#
```

Assigning OSPFv3 areas in a non-default VRF

Areas can be assigned as OSPFv3 areas in a non-default VRF.

Enable IPv6 on each interface on which you plan to enable OSPFv3. You enable IPv6 on an interface by configuring an IP address or explicitly enabling IPv6 on that interface.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **vrf** command and specify a name to enter Virtual Routing and Forwarding (VRF) configuration mode and create a non-default VRF instance.

   ```
   device(config)# vrf red
   ```

3. Enter the **rd** command, assigning an administrative number and arbitrary number the route, to distinguish a route for VRF green.

   ```
   device(config-vrf-red)# rd 100:200
   ```

4. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config-vrf-red)# ip router-id 10.11.12.14
   ```

5. Enter the **address-family ipv6** command to enter IPv6 address-family configuration mode.

```
device(config-vrf-red)# address-family ipv6
```

6. Enter the **exit** command until you return to global configuration mode.

```
device(config-vrf-red-ipv6)# exit
```

7. Enter the **ipv6 router ospf** command and specify a VRF name to enter OSPFv3 configuration mode and enable OSPFv3 in a non-default VRF.

```
device(config)# ipv6 router ospf vrf red
```

8. Enter the **area** command to define an OSPFv3 area ID.

```
device(config-ospf6-router-vrf-red)# area 0
```

9. Enter the **area** command to define a second OSPFv3 area ID.

```
device(config-ospf6-router-vrf-red)# area 10.1.1.1
```

The following example assigns an OSPFv3 ID to two areas in a non-default VRF instance. One of the areas is assigned by decimal number. The second area is assigned by IP address.

```
device# configure terminal
device(config)# vrf red
device(config-vrf-red)# rd 100:200
device(config-vrf-red)# ip router-id 10.11.12.13
device(config-vrf-red)# address-family ipv6
device(config-vrf-red-ipv6)#
device(config-vrf-red-ipv6)# exit
device(config)# ipv6 router ospf vrf red
device(config-ospf6-router-vrf-red)# area 0
device(config-ospf6-router-vrf-red)# area 10.1.1.1
```

Assigning OSPFv3 areas in a non-default VRF

Areas can be assigned as OSPFv3 areas in a non-default VRF.

Enable IPv6 on each interface on which you plan to enable OSPFv3. You enable IPv6 on an interface by configuring an IP address or explicitly enabling IPv6 on that interface.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **vrf** command and specify a name to enter Virtual Routing and Forwarding (VRF) configuration mode and create a non-default VRF instance.

```
device(config-rbridge-id-122)# vrf red
```

4. Enter the **ip router-id** command to specify the router ID.

```
device(config-vrf-red)# ip router-id 10.11.12.14
```

5. Enter the **address-family ipv6 unicast** command to enter IPv6 address-family configuration mode.

```
device(config-vrf-red)# address-family ipv6 unicast
```

6. Enter the **exit** command until you return to RBridge ID configuration mode.

```
device(vrf-ipv6-unicast)# exit
```

7. Enter the **ipv6 router ospf** command and specify a VRF name to enter OSPFv3 configuration mode and enable OSPFv3 in a non-default VRF.

```
device(config-rbridge-id-122)# ipv6 router ospf vrf red
```

8. Enter the **area** command to define an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-red)# area 0
```

9. Enter the **area** command to define a second OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-red)# area 10.1.1.1
```

The following example assigns an OSPFv3 ID to two areas in a non-default VRF instance. One of the areas is assigned by decimal number. The second area is assigned by IP address.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# vrf red
device(config-vrf-red)# ip router-id 10.11.12.13
device(config-vrf-red)# address-family ipv6 unicast
device(vrf-ipv6-unicast)#
device(vrf-ipv6-unicast)# exit
device(config-vrf-red)# exit
device(config-rbridge-id-122)# ipv6 router ospf vrf red
device(config-ipv6-router-ospf-vrf-red)# area 0
device(config-ipv6-router-ospf-vrf-red)# area 10.1.1.1
```

Setting all OSPFv3 interfaces to the passive state

All OSPFv3 interfaces can be set as passive, causing them to drop all OSPFv3 control packets.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config)# ipv6 router ospf
```

```
device(config-rbridge-id-122)# ipv6 router ospf
```

4. Enter the **default-passive-interface** command to mark all interfaces passive by default.

```
device(config-ospf6-router)# default-passive-interface
```

```
device(config-ipv6-router-ospf-vrf-default-vrf)# default-passive-interface
```

The following example sets all OSPFv3 interfaces as passive, causing them to drop all the OSPFv3 control packets.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# default-passive-interface
```

The following example sets all OSPFv3 interfaces for a specified RBridge as passive, causing them to drop all the OSPFv3 control packets.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# default-passive-interface
```

The following example sets all OSPFv3 interfaces as passive, causing them to drop all the OSPFv3 control packets.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# default-passive-interface
```

OSPFv3 graceful restart helper

The OSPFv3 graceful restart (GR) helper provides a device with the capability to participate in a graceful restart in helper mode so that it assists a neighboring routing device that is performing a graceful restart.

When OSPFv3 GR helper is enabled on a device, the device enters helper mode upon receipt of a grace-LSA where the neighbor state is full. By default, the helper capability is enabled when you start OSPFv3, even if graceful restart is not supported.

> **NOTE**
> Process restart takes precedence over GR.

Disabling OSPFv3 graceful restart helper

The OSPFv3 graceful restart (GR) helper is enabled by default, and can be disabled on a routing device.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

4. Enter the **no graceful-restart helper** command with the **strict-lsa-checking** to disable the GR helper with strict link-state advertisement (LSA) checking.

   ```
   device(config-ospf6-router)# no graceful-restart helper strict-lsa-checking
   device(config-ipv6-router-ospf-vrf-default-vrf)# no graceful-restart helper strict-lsa-checking
   ```

5. Enter the **no graceful-restart helper** command to disable the GR helper.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# no graceful-restart helper
   ```

The following example disables the GR helper with strict link-state advertisement (LSA) checking.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# no graceful-restart helper strict-lsa-checking

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# no graceful-restart helper strict-lsa-checking
```

The following example disables the GR helper.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# no graceful-restart helper
```

Re-enabling OSPFv3 graceful restart helper

If the OSPFv3 graceful restart (GR) helper has been disabled on a routing device, it can be re-enabled. GR helper mode can also be enabled with strict link-state advertisement (LSA) checking.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf

   device(config-rbridge-id-122)# ipv6 router ospf
   ```

4. Enter the **graceful-restart helper** command and specify the **strict-lsa-checking** parameter to re-enable the GR helper with strict LSA checking.

   ```
   device(config-ospf6-router)# graceful-restart helper strict-lsa-checking

   device(config-ipv6-router-ospf-vrf-default-vrf)# graceful-restart helper strict-lsa-checking
   ```

The following example re-enables the GR helper with strict LSA checking.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# graceful-restart helper strict-lsa-checking

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# graceful-restart helper strict-lsa-checking

device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# graceful-restart helper strict-lsa-checking
```

OSPFv3 non-stop routing

OSPFv3 can continue operation without interruption during hitless failover when the NSR feature is enabled.

> **NOTE**
> IS-IS NSR is supported only on the SLX 9850 with redundant MMs.

During graceful restart (GR), the restarting neighbors must help build routing information during a failover. However, the GR helper may not be supported by all devices in a network. Non-stop routing (NSR) eliminates this dependency.

NSR does not require support from neighboring devices to perform hitless failover, and OSPF can continue operation without interruption.

> **NOTE**
> Process restart takes precedence over NSR.

> **NOTE**
> NSR does not support IPv6-over-IPv4 tunnels and virtual links, so traffic loss is expected while performing hitless failover.

> **NOTE**
> NSR does not support virtual links, so traffic loss is expected while performing hitless failover.

Enabling OSPFv3 NSR

OSPFv3 non-stop routing (NSR) can be re-enabled if it has been disabled. The following task re-enables NSR for OSPFv3.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

4. Enter the **graceful restart** command to re-enable GR on the device.

   ```
   device(config-ospf6-router)# nonstop-routing
   device(config-ipv6-router-ospf-vrf-default-vrf)# nonstop-routing
   ```

The following example re-enables NSR for OSPFv3.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# nonstop-routing

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# nonstop-routing

device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# nonstop-routing
```

OSPFv3 max-metric router LSA

OSPFv3 can be configured to advertise its locally generated router LSAs with a maximum metric to direct transit traffic away from the device, while still routing for directly connected networks.

By advertising the maximum metric, the device does not attract transit traffic. A device which does not handle transit traffic, and only forwards packets destined for its directly connected links, is known as a stub router. In OSPFv3 networks, a device could be placed in a stub router role by advertising large metrics for its connected links, so that the cost of a path through the device becomes larger than that of an alternative path.

You can configure OSPFv3 max-metric router LSA in either startup or non-startup mode. Configuring max-metric on startup may be helpful on ASBRs where protocols such as BGP converge after OSPF converges. Configuring max-metric on non-startup may be helpful in database overflow scenarios.

> NOTE
> The **on-startup** configuration does not apply to NSR restarts.

Configuring the OSPFv3 max-metric router LSA

By configuring the OSPFv3 max-metric router LSA feature you can enable OSPFv3 to advertise its locally generated router LSAs with a maximum metric.

> NOTE
> You can configure OSPFv3 max-metric router LSA in either startup or non-startup mode. Configuring max-metric with non-startup mode, it is only applied once and is not persistent across reloads, or after the **clear ip ospf all** command is issued.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config)# ip router-id 10.11.12.13
   device(config-rbridge-id-122)# ip router-id 10.11.12.13
   ```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

```
device(config-rbridge-id-122)# ipv6 router ospf
```

5. Enter the **max-metric router-lsa** command with the **external-lsa** keyword and specify a value to configure the maximum metric value .

```
device(config-ospf6-router)# max-metric router-lsa external-lsa 1500
```

```
device(config-ipv6-router-ospf-vrf-default-vrf)# max-metric router-lsa external-lsa 1500
```

6. Enter the **max-metric router-lsa** command with the **on-startup** keyword and specify a value to specify a period of time to advertise a maximum metric after a restart before advertising with a normal metric.

```
device(config-ospf6-router)# max-metric router-lsa on-startup 85
```

```
device(config-ipv6-router-ospf-vrf-default-vrf)# max-metric router-lsa on-startup 85
```

The following example configures an OSPFv3 device to advertise a maximum metric and sets the maximum metric value for external LSAs to 1500, and configures the device to advertise a maximum metric for 85 seconds after a restart before advertising with a normal metric.

```
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 router ospf
device(config-ospf6-router)# max-metric router-lsa external-lsa 1500
device(config-ospf6-router)# max-metric router-lsa on-startup 85

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.11.12.13
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# max-metric router-lsa external-lsa 1500
device(config-ipv6-router-ospf-vrf-default-vrf)# max-metric router-lsa on-startup 85

device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# max-metric router-lsa external-lsa 1500
device(config-ipv6-router-ospf-vrf-default-vrf)# max-metric router-lsa on-startup 85
```

IPsec for OSPFv3

IP Security (IPsec) secures OSPFv3 communications by authenticating and encrypting each IP packet of a communication session.

IPsec provides security features such as data integrity, replay protection, and message confidentiality. You can use IPsec to secure specific OSPFv3 areas and interfaces and protect OSPFv3 virtual links.

IPsec provides security features such as authentication of data origin, data integrity, replay protection, and message confidentiality. You can use IPsec to secure specific OSPFv3 areas and interfaces and protect OSPFv3 virtual links.

The Encapsulating Security Payload (ESP) protocol authenticates routing information between peers. ESP can provide message confidentiality, connectionless data integrity, and optional replay protection. ESP has both a header and a trailer. The authentication data of ESP cannot protect the outer IP header, only the payload that is being encrypted.

IPsec for OSPFv3 constitutes two basic protocols to authenticate routing information between peers:

- **Authentication header (AH)**: AH provides data origin authentication and connectionless integrity, as well as providing the optional replay protection feature. AH authenticates as much of the IP header as possible, as well as the upper-level protocol

data such as source IPv6 address, destination IPv6 address, flags, and IP payload. However, some IP header fields, such as TTL and checksum are often modified in transit and, therefore, cannot be protected by AH.

- **Encapsulating Security Payload (ESP)**: ESP can provide message confidentiality, connectionless data integrity, and optional replay protection. ESP has both a header and a trailer. The authentication data of ESP cannot protect the outer IP header, only the payload that is being encrypted.

IPsec is available for OSPFv3 traffic only and only for packets that are “for-us”. A for-us packet is addressed to one of the IPv6 addresses on the device or to an IPv6 multicast address. Packets that are only forwarded by the line card do not receive IPsec scrutiny.

IPsec is available for OSPFv3 traffic only and only for packets that are “for-us”. A for-us packet is addressed to one of the IPv6 addresses on the device or to an IPv6 multicast address.

The devices support the following components of IPsec for IPv6-addressed packets:

- Authentication through AH
- Authentication through ESP in transport mode
- Hashed Message Authentication Code-Secure Hash Algorithm 1 (HMAC-SHA-1) as the authentication algorithm
- Hashed Message Authentication Code-Message Digest 5 (HMAC-MD5) as the authentication algorithm
- Security parameter index (SPI)
- Manual configuration of keys
- Configurable rollover timer

IPsec can be enabled on the following logical entities:

- Interface
- Area
- Virtual link

IPsec is based on security associations (SAs). With respect to traffic classes, this implementation of IPsec uses a single security association between the source and destination to support all traffic classes and does not differentiate between the different classes of traffic that the DSCP bits define.

IPsec on a virtual link is a global configuration. Interface and area IPsec configurations are more granular.

Among the entities that can have IPsec protection, the interfaces and areas can overlap. The interface IPsec configuration takes precedence over the area IPsec configuration when an area and an interface within that area use IPsec. Therefore, if you configure IPsec for an interface and an area configuration also exists that includes this interface, the interface’s IPsec configuration is used by that interface. However, if you disable IPsec on an interface, IPsec is disabled on the interface even if the interface has its own specific authentication.

For IPsec, the system generates two types of databases. The Security Association Database (SAD) contains a security association for each interface or one global database for a virtual link. Even if IPsec is configured for an area, each interface that uses the area’s IPsec still has its own security association in the SAD. Each SA in the SAD is a generated entry that is based on your specifications of an authentication protocol (for example, ESP), destination address, and a security parameter index (SPI). The SPI number is user-specified according to the network plan. Consideration for the SPI values to specify must apply to the whole network.

For IPsec, the system generates two types of databases. The Security Association Database (SAD) contains a security association for each interface or one global database for a virtual link. Even if IPsec is configured for an area, each interface that uses the area’s IPsec still has its own security association in the SAD. Each SA in the SAD is a generated entry that is based on your specifications of an authentication protocol, destination address, and a security parameter index (SPI). The SPI number is user-specified according to the network plan. Consideration for the SPI values to specify must apply to the whole network.

The system-generated security policy databases (SPDs) contain the security policies against which the system checks the for-us packets. For each for-us packet that has an ESP header, the applicable security policy in the security policy database (SPD) is checked to see if

this packet complies with the policy. The IPsec task drops the non-compliant packets. Compliant packets continue on to the OSPFv3 task.

IPsec for OSPFv3 configuration

IPsec authentication can be enabled on both default and nondefault VRFs. IPsec authentication is disabled by default.

The following IPsec parameters are configurable:

- AH security protocol
- ESP protocol
- Authentication
- Hashed Message Authentication Code-Message Digest 5 (HMAC-MD5) authentication algorithm
- Hashed Message Authentication Code-Secure Hash Algorithm 1 (HMAC-SHA-1) authentication algorithm
- Security parameter index (SPI)
- A 40-character key using hexadecimal characters
- An option for not encrypting the keyword when it appears in **show** command output
- Key rollover timer
- Specifying the key add remove timer

IPsec for OSPFv3 considerations

IPsec generates security associations and security policies based on certain user-specified parameters. Refer to the *NetIron Command Reference* for more information on user-specified parameters.

- The system creates a security association for each interface or virtual link based on the values specified by the user.
- The system creates a security policy database for each interface or virtual link based on the values specified by the user.
- You can configure the same SPI and key on multiple interfaces and areas, but they still have unique IPsec configurations because the SA and policies are added to each separate security policy database (SPD) that is associated with a particular interface. If you configure an SA with the same SPI in multiple places, the rest of the parameters associated with the SA—such as key, cryptographic algorithm, security protocol, and so on—must match. If the system detects a mismatch, it displays an error message.
- IPsec authentication for OSPFv3 requires the use of multiple SPDs, one for each interface. A virtual link has a separate, global SPD. The authentication configuration on a virtual link must be different from the authentication configuration for an area or interface, as required by RFC4 552. The interface number is used to generate a non-zero security policy database identifier (SPDID), but for the global SPD for a virtual link, the system-generated SPDID is always zero. As a hypothetical example, the SPD for interface eth 1/1 might have the system-generated SPDID of 1, and so on.
- If you change an existing key, you must also specify a different SPI value. For example, in an interface context where you intend to change a key, you must enter a different SPI value—which occurs before the key parameter on the command line—before you enter the new key.
- The old key is active for twice the current configured key rollover interval for the inbound direction. In the outbound direction, the old key remains active for a duration equal to the key rollover interval. If the key rollover interval is set to 0, the new key immediately takes effect for both directions.

Configuring IPsec on an OSPFv3 area

IPsec can be configured to secure communications on an OSPFv3 area.

Currently certain keyword parameters must be entered though only one keyword choice is possible for that parameter. For example, the only authentication algorithm is HMAC-SHA1-96, but you must nevertheless enter the **sha1** keyword for this algorithm. Also, although ESP is currently the only authentication protocol, you must enter the **esp** keyword.

> NOTE
> When IPsec is configured for an area, the security policy is applied to all the interfaces in the area.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config)# ip router-id 10.11.12.13
   device(config-rbridge-id-122)# ip router-id 10.11.12.13
   ```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 configuration mode and enable OSPFv3 on the device.

   ```
   device(config)# ipv6 router ospf
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

5. Enter **area authentication ipsec spi** *spi* **esp sha1**, specifying an area, and enter a 40-character hexadecimal key.

   ```
   device(config-ospf6-router)# area 0 authentication ipsec spi 600 esp sha1
   abcef12345678901234fedcba098765432109876
   ```

   IPsec is configured in OSPv3 area 0 with a security parameter index (SPI) value of 600, and Hashed Message Authentication Code (HMAC) Secure Hash Algorithm 1 (SHA-1) authentication is enabled.

6. Enter **area authentication spi** *spi* **ah hmac-md5 key**, specifying an area, and enter a 40-character hexadecimal key.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# area 0 authentication spi 600 ah hmac-md5 key
   abcef12345678901234fedcba098765432109876
   ```

   IPsec is configured in OSPv3 area 0 with a security parameter index (SPI) value of 600, and the authentication header (AH) protocol is selected. Message Digest 5 (MD5) authentication on the area is enabled.

The following example enables HMAC SHA-1 authentication for the OSPFv3 area, setting an SPI value of 600.

```
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 router ospf
device(config-ospf6-router)# area 0 authentication ipsec spi 600 esp sha1
abcef12345678901234fedcba098765432109876
```

The following example enables AH and MD5 authentication for the OSPFv3 area, setting an SPI value of 600.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.11.12.13
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 0 authentication spi 600 ah hmac-md5 key
abcef12345678901234fedcba098765432109876

device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 0 authentication spi 600 ah hmac-md5 key
abcef12345678901234fedcba098765432109876
```

Configuring IPsec on an OSPFv3 interface

IPsec can be configured to secure communications on an OSPFv3 interface.

For IPsec to work, the IPsec configuration must be the same on all the routers to which an interface connects.

Currently certain keyword parameters must be entered though only one keyword choice is possible for that parameter. For example, the only authentication algorithm is HMAC-SHA1-96, but you must nevertheless enter the **sha1** keyword for this algorithm. Also, although ESP is currently the only authentication protocol, you must enter the **esp** keyword.

> NOTE
> Ensure that OSPFv3 areas are assigned. All device interfaces must be assigned to one of the defined areas on an OSPFv3 router. When an interface is assigned to an area, all corresponding subnets on that interface are automatically included in the assignment.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface** command and specify an interface.

   ```
   device(config)# interface ethernet 1/1
   device(config)# interface ethernet 0/1
   device(config)# interface tengigabitethernet 1/0/1
   ```

3. Enter the **ipv6 ospf area** command to assign a specified area to the interface.

   ```
   device(config-if-e1000/1/1)# ipv6 ospf area 0
   device(config-if-te-1/0/1)# ipv6 ospf area 0
   device(conf-if-eth-1/1)# ipv6 ospf area 0
   device(conf-if-eth-0/1)# ipv6 ospf area 0
   ```

4. Enter **ipv6 ospf authentication ipsec spi** *value* **esp sha1** and specify a 40-character hexadecimal key.

   ```
   device(config-if-e1000/1/1)# ipv6 ospf authentication ipsec spi 512 esp sha1
   abcef12345678901234fedcba098765432109876
   ```

   IPsec is configured on the specified interface with a security parameter index (SPI) value of 512, and the Encapsulating Security Payload (ESP) protocol is selected. Secure Hash Algorithm 1 (SHA-1) authentication is enabled.

5. Enter **ipv6 ospf authentication spi** *value* **esp null hmac-sha1** and specify a 40-character hexadecimal key.

```
device(config-if-te-1/0/1)# ipv6 ospf authentication spi 512 esp null hmac-sha1 key
abcef12345678901234fedcba098765432109876

device(conf-if-eth-1/1)# ipv6 ospf authentication spi 512 esp null hmac-sha1 key
abcef12345678901234fedcba098765432109876

device(conf-if-eth-0/1)# ipv6 ospf authentication spi 512 esp null hmac-sha1 key
abcef12345678901234fedcba098765432109876
```

IPsec is configured on the specified interface with a security parameter index (SPI) value of 512, and the Encapsulating Security Payload (ESP) protocol is selected. Secure Hash Algorithm 1 (SHA-1) authentication is enabled.

The following example enables ESP and SHA-1 on a specified OSPFv3 Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/1
device(config-if-e1000/1/1)# ipv6 ospf area 0
device(config-if-e1000/1/1)# ipv6 ospf authentication ipsec spi 512 esp sha1
abcef12345678901234fedcba098765432109876

device# configure terminal
device(config)# interface ethernet 1/1
device(conf-if-eth-1/1)# ipv6 ospf area 0
device(conf-if-eth-1/1)# ipv6 ospf authentication spi 512 esp null hmac-sha1 key
abcef12345678901234fedcba098765432109876

device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# ipv6 ospf area 0
device(conf-if-eth-0/1)# ipv6 ospf authentication spi 512 esp null hmac-sha1 key
abcef12345678901234fedcba098765432109876
```

The following example enables ESP and SHA-1 on a specified OSPFv3 10-gigabit interface.

```
device# configure terminal
device(config)# interface tengigabitethernet 1/0/1
device(config-if-te-1/0/1)# ipv6 ospf area 0
device(config-if-te-1/0/1)# ipv6 ospf authentication spi 512 esp null hmac-sha1 key
abcef12345678901234fedcba098765432109876
```

Configuring IPsec on OSPFv3 virtual links

IP Security (IPsec) can be configured for virtual links.

An OSPFv3 virtual link must be configured.

Currently certain keyword parameters must be entered though only one keyword choice is possible for that parameter. For example, the only authentication algorithm is HMAC-SHA1-96, but you must nevertheless enter the **sha1** keyword for this algorithm. Also, although ESP is currently the only authentication protocol, you must enter the **esp** keyword.

The virtual link IPsec security associations (SAs) and policies are added to all interfaces of the transit area for the outbound direction. For the inbound direction, IPsec SAs and policies for virtual links are added to the global database.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ip router-id** command to specify the router ID.

```
device(config)# ip router-id 10.1.1.1
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

4. Enter **area virtual-link authentication ipsec spi** *value* **esp sha1 no-encrypt** *key*, specifying an area address and the ID of the OSPFv3 device at the remote end of the virtual link..

```
device(config-ospf6-router)# area 1 virtual-link 10.1.1.1 authentication ipsec spi 512 esp sha1 no-
encrypt 1134567890223456789012345678901234567890
```

IPsec is configured on the specified virtual link in OSPF area 1.The device ID associated with the virtual link neighbor is 10.1.1.1, the SPI value is 512, and the Encapsulating Security Payload (ESP) protocol is selected. Secure Hash Algorithm 1 (SHA-1) authentication is enabled. The 40-character key is not encrypted in **show** command displays.

The following example configures IPsec on an OSPFv3 area.

```
device# configure terminal
device(config)# ip router-id 10.1.1.1
device(config)# ipv6 router ospf
device(config-ospf6-router)# area 1 virtual-link 10.1.1.1 authentication ipsec spi 512 esp sha1 no-encrypt
1134567890223456789012345678901234567890
```

Configuring IPsec on OSPFv3 virtual links

IP Security (IPsec) can be configured for virtual links.

An OSPFv3 virtual link must be configured.

The virtual link IPsec security associations (SAs) and policies are added to all interfaces of the transit area for the outbound direction. For the inbound direction, IPsec SAs and policies for virtual links are added to the global database.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip router-id** command to specify the router ID.

```
device(config-rbridge-id-122)# ip router-id 10.1.1.1
```

```
device(config)# ip router-id 10.1.1.1
```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 configuration mode and enable OSPFv3 on the router.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

```
device(config)# ipv6 router ospf
```

5. Enter **area virtual-link authentication spi** *value* **ah hmac-sha1 no-encrypt** *key*, specifying an area address and the ID of the OSPFv3 device at the remote end of the virtual link.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1 authentication spi 512
ah hmac-sha1 no-encrypt key 1134567890223456789012345678901234567890
```

IPsec is configured on the specified virtual link in OSPF area 1. The device ID associated with the virtual link neighbor is 10.1.1.1, the SPI value is 512, and the authentication header (AH) protocol is selected. Secure Hash Algorithm 1 (SHA-1) authentication is enabled. The 40-character key is not encrypted in **show** command displays.

6. Enter **area virtual-link authentication spi** *value* **ah hmac-sha1** *key*, specifying an area address and the ID of the OSPFv3 device at the remote end of the virtual link.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1 authentication spi 512
ah hmac-sha1 key 1134567890223456789012345678901234567890
```

   IPsec is configured on the specified virtual link in OSPF area 1. The device ID associated with the virtual link neighbor is 10.1.1.1, the SPI value is 512, and the authentication header (AH) protocol is selected. Secure Hash Algorithm 1 (SHA-1) authentication is enabled. The 40-character key is not encrypted in **show** command displays.

The following example configures IPsec on an OSPFv3 area.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.1.1.1
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1 authentication spi 512 ah
hmac-sha1 no-encrypt key 1134567890223456789012345678901234567890

device# configure terminal
device(config)# ip router-id 10.1.1.1
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1 authentication spi 512 ah
hmac-sha1 key 1134567890223456789012345678901234567890
```

Specifying the key rollover and key add-remove timers

The key rollover timer can be configured so that rekeying takes place on all the nodes at the same time and the security parameters are consistent across all the nodes. The timing of the authentication key add-remove interval can also be altered.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip router-id** command to specify the router ID.

```
device(config)# ip router-id 10.11.12.13
```

```
device(config-rbridge-id-122)# ip router-id 10.11.12.13
```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

```
device(config-rbridge-id-122)# ipv6 router ospf
```

5. Enter the **key-add-remove-interval** command and specify the desired interval to set the timing of the authentication key add-remove interval.

```
device(config-ospf6-router)# key-add-remove-interval 240
```

```
device(config-ipv6-router-ospf-vrf-default-vrf)# key-add-remove-interval 240
```

6. Enter the **key-rollover-interval** command and specify the desired interval to set the timing of the configuration changeover.

```
device(config-ospf6-router)# key-rollover-interval 240
```

```
device(config-ipv6-router-ospf-vrf-default-vrf)# key-rollover-interval 240
```

The following example sets the key add-remove interval to 240 seconds (4 minutes) and sets the timing of the configuration changeover to 240 seconds (4 minutes).

```
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 router ospf
device(config-ospf6-router)# key-add-remove-interval 240
device(config-ospf6-router)# key-rollover-interval 240

device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.11.12.13
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# key-add-remove-interval 240
device(config-ipv6-router-ospf-vrf-default-vrf)# key-rollover-interval 240

device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# key-add-remove-interval 240
device(config-ipv6-router-ospf-vrf-default-vrf)# key-rollover-interval 240
```

Clearing IPsec statistics

Statistics related to IP security (IPsec) can be cleared using the **clear ipsec statistics** command.

Use the **show ipsec statistics** command to display the IPsec statistics. After using the **clear ipsec statistics** command to clear the IPsec statistics, re-enter the **show ipsec statistics** command to verify the IPsec statistics have been cleared. **The clear ipsec statistics** command resets the IPsec packet statistics and IPsec error statistics counters to zero.

1. Enter the **exit** command as necessary to access user EXEC mode

```
device(config)# exit
```

2. Enter the **show ipsec statistics** command to display statistics related to IPsec.

```
device# show ipsec statistics
                      IPSecurity Statistics
secEspCurrentInboundSAs 1          ipsecEspTotalInboundSAs:  2
secEspCurrentOutboundSA 1          ipsecEspTotalOutboundSAs: 2
                    IPSecurity Packet Statistics
secEspTotalInPkts:      20         ipsecEspTotalInPktsDrop:  0
secEspTotalOutPkts:     84
                    IPSecurity Error Statistics
secAuthenticationErrors 0
secReplayErrors:        0          ipsecPolicyErrors:        13
secOtherReceiveErrors:  0          ipsecSendErrors:          0
secUnknownSpiErrors:    0
```

3. Enter the **clear ipsec statistics** command to clear statistics related to IPsec from the configuration.

```
device# clear ipsec statistics
```

4. Enter the **show ipsec statistics** command to verify that statistics related to IPsec have been cleared from the configuration.

```
device# show ipsec statistics
                        IPSecurity Statistics
ipsecEspCurrentInboundSAs 0          ipsecEspTotalInboundSAs:  0
ipsecEspCurrentOutboundSA 0          ipsecEspTotalOutboundSAs: 0
                       IPSecurity Packet Statistics
ipsecEspTotalInPkts:      0          ipsecEspTotalInPktsDrop:  0
ipsecEspTotalOutPkts:     0
                       IPSecurity Error Statistics
ipsecAuthenticationErrors 0
ipsecReplayErrors:        0          ipsecPolicyErrors:        0
ipsecOtherReceiveErrors:  0          ipsecSendErrors:          0
ipsecUnknownSpiErrors:    0
device#
```

The counters holding IPsec packet statistics and IPsec error statistics are reset to 0.

The following example clears IPsec statistics and verifies that the IPsec statistics have been cleared.

```
device(config-ospf6-router)# exit
device(config)# exit
device# show ipsec statistics
device# clear ipsec statistics
device# show ipsec statistics
```

Displaying OSPFv3 results

The **show ipv6 ospf** command and its variations can be used to display information about OSPFv3 configurations.

Use one or more of the following commands to verify OSPFv3 information. Using the **show ipv6 ospf** command is optional, and the variations of the command can be entered in any order.

1. Enter the **exit** command as necessary to access user EXEC mode.

```
device# exit
```

2. Enter the **show ipv6 ospf** command to display general OSPFv3 information.

```
device> show ipv6 ospf

OSPFv3 Process number 0 with Router ID 0x10010101(10.1.1.1)
    Running 0 days 0 hours 1 minutes 53 seconds
    Number of AS scoped LSAs is 3
    Sum of AS scoped LSAs Checksum is fabdd4de
 External LSA Limit is 250000
 Database Overflow Interval is 10
 Database Overflow State is NOT OVERFLOWED
 Route calculation executed 0 times
 Pending outgoing LSA count 0
 Authentication key rollover interval 30 seconds
    Authentication key add/remove interval 0 seconds
 Number of areas in this router is 3
 Router is operating as ABR
 Router is operating as ASBR, Redistribute: CONNECTED
 High Priority Message Queue Full count: 0
 BFD is disabled
```

3. The following example of the **show ipv6 ospf area** command shows detailed output for assigned OSPFv3 Area 0.

```
device> show ipv6 ospf area 0

Area 0:
  Interface attached to this area: loopback 2 ethe 3/2 tunnel 2
        Number of Area scoped LSAs is 6
        Statistics of Area 0:
            SPF algorithm executed 16 times
            SPF last updated: 335256 sec ago
            Current SPF node count: 3
                Router: 2 Network: 1
                Maximum of Hop count to nodes: 2
```

4. The following example of the **show ipv6 ospf interface brief** command shows limited OSPFv3 interface information.

```
device> show ipv6 ospf interface brief

Interface     Area     Status Type Cost  State    Nbrs(F/C)
eth 1/1       1        up      BCST  1     BDR      0/1
eth 2/1       1        up      BCST  1     DR       0/0
loopback 1    1        up      BCST  1     Loopback 0/0
```

5. The following example of the **show ipv6 ospf neighbor** command shows OSPFv3 neighbor information for the device.

```
device> show ipv6 ospf neighbor

RouterID     Pri State    DR              BDR         Interface [State]
10.1.1.1     1 Full       10.223.223.223  10.1.1.1    ethe 3/2  [DR]
```

6. The following example of the **show ipv6 ospf virtual-neighbor** command shows information about an OSPFv3 virtual neighbor.

```
device> show ipv6 ospf virtual-neighbor

Index Router ID        Address                       State     Interface
1     10.14.14.14      2001:db8:44:44::4             Full      eth 1/8
                Option: 00-00-00    QCount: 0    Timer: 408
2     10.14.14.14      2001:db8:44:44::4             Full      tunnel 256
                Option: 00-00-00    QCount: 0    Timer: 43
```

7. The following example of the **show ipv6 ospf database** command shows information about different OSPFv3 LSAs.

```
device> show ipv6 ospf database

LSA Key - Rtr:Router Net:Network Inap:InterPrefix Inar:InterRouter
          Extn:ASExternal Grp:GroupMembership Typ7:Type7 Link:Link
          Iap:IntraPrefix Grc:Grace
Area ID     Type LSID     Adv Rtr      Seq(Hex)  Age  Cksum Len   Sync
0           Iap  0        10.1.1.1     80000001  3    343d  52    Yes
0           Iap  0        10.2.2.2     80000001  8    c61d  58    No
```

8. The following example of the **show ipv6 ospf routes** command shows output for OSPFv3 routes.

```
device> show ipv6 ospf routes

Current Route count: 4
    Intra: 4 Inter: 0 External: 0 (Type1 0/Type2 0)
    Equal-cost multi-path: 0
    OSPF Type: IA- Intra, OA - Inter, E1 - External Type1, E2 - External Type2
   Destination                Cost     E2Cost    Tag        Flags    Dis
IA 2001:db8:200:1::1/128      0        0         0          00000003 110
   Next_Hop_Router            Outgoing_Interface Adv_Router
   ::                         loopback 1         10.1.2.1
   Destination                Cost     E2Cost    Tag        Flags    Dis
IA 2001:db8:300:1::1/128      0        0         0          00000003 110
   Next_Hop_Router            Outgoing_Interface Adv_Router
   ::                         loopback 2         10.1.2.1
   Destination                Cost     E2Cost    Tag        Flags    Dis
IA 2001:db8:400:1::1/128      0        0         0          00000003 110
   Next_Hop_Router            Outgoing_Interface Adv_Router
   ::                         loopback 1         10.1.2.1
   Destination                Cost      E2Cost   Tag        Flags    Dis
IA 2001:db8:500:1::1/128      1         0        0          00000003 110
   Next_Hop_Router            Outgoing_Interface Adv_Router
   ::                         loopback 2         10.1.2.1
```

9. The following example of the **show ipv6 ospf database** command with the **tree** shows information about the SPF trees.

```
device> show ipv6 ospf spf tree

 SPF tree for Area 0
+- 10.223.223.223 cost 0
+- 10.223.223.223:88 cost 1
+- 10.1.1.1:0 cost 1
```

10. The following example of the **show ipv6 ospf database** command with the **table** shows information about the SPF table.

```
device> show ipv6 ospf spf table

SPF table for Area 0
 Destination         Bits Options  Cost Nexthop                   Interface
R 10.1.1.1           ---- V6E---R-   1  fe80::2e0:52ff:fe91:bb37  ethe 3/2
N 10.223.223.223[88] ---- V6E---R-   1  ::                         ethe 3/2
```

11. The following example of the **show ipv6 ospf redistribute route** command shows information about routes that the device has redistributed into OSPFv3.

```
device> show ipv6 ospf redistribute route

Id     Prefix                  Protocol  Metric Type  Metric
1      2001:db8::/32           Static    Type-2       1
2      2001:db8:1234::/48      Static    Type-2       0
```

12. The following example of the **show ipv6 ospf routes** command shows information about a specified OSPFv3 route.

```
device> show ipv6 ospf routes 2000::
Destination                 Cost      E2Cost    Tag       Flags    Dis
IA 2000::/64                1         0         0         00000003 110
   Next_Hop_Router           Outgoing_Interface Adv_Router
   ::                          eth 1/1            10.1.1.1
```

Displaying OSPFv3 results

The **show ipv6 ospf** command and its variations can be used to display information about OSPFv3 configurations.

Use one or more of the following commands to verify OSPFv3 information. Using the **show ipv6 ospf** command is optional, and the variations of the command can be entered in any order.

1. Enter the **exit** command as necessary to access privileged EXEC mode.

```
device(config)# exit
```

2. Enter the **show ipv6 ospf** command to display general OSPFv3 information.

```
device# show ipv6 ospf

OSPFv3 Process number 0 with Router ID 0x01010101(1.1.1.1)
 Running 0 days 0 hours 24 minutes 36 seconds
 Number of AS scoped LSAs is 0
 Sum of AS scoped LSAs Checksum is 00000000
 External LSA Limit is 250000
 Database Overflow Interval is 10
 Database Overflow State is NOT OVERFLOWED
 Route calculation executed 34 times
 Pending outgoing LSA count 0
 Authentication key rollover interval 300 seconds
 Number of areas in this router is 2
 Router is operating as ABR
 High Priority Message Queue Full count: 0
 Graceful restart helper is enabled, strict lsa checking is disabled
 Nonstop Routing is disabled
```

   The example output offers general OSPFv3 information and indicates that the device is not operating as an ASBR. If the device is not operating as an ASBR, there is no information about redistribution in the output.

3. The following example of the **show ipv6 ospf summary** command shows summary output for the one IPv6 OSPF session that is configured.

```
device# show ipv6 ospf summary

Total number of IPv6 OSPF instances: 1
Seq Instance        Intfs   Nbrs    Nbrs-Full LSAs    Routes
1   default-vrf     3       2       2         25      6
```

4. The following example of the **show ipv6 ospf area** command shows detailed output for assigned OSPFv3 Area 0.

```
device# show ipv6 ospf area 0

Area 0:
  Authentication: Not Configured
  Active interface(s)attached to this area: Te 1/0/1 VLink 1
  Inactive interface(s)attached to this area: None
  Number of Area scoped LSAs is 12
  Sum of Area LSAs Checksum is 0006e83c
  Statistics of Area 0:
    SPF algorithm executed 17 times
    SPF last updated: 333 sec ago
    Current SPF node count: 2
      Router: 2 Network: 0
      Maximum of Hop count to nodes: 1
```

5. The following example of the **show ipv6 ospf interface brief** command shows limited OSPFv3 interface information.

```
device# show ipv6 ospf interface brief
Interface     Area     Status Type Cost  State    Nbrs(F/C)
Te 1/0/1      0        up     BCST 1     DR       0/0
Te 1/0/2      1        up     BCST 1     BDR      1/1
```

6. The following example of the **show ipv6 ospf neighbor** command shows OSPFv3 neighbor information for the device.

```
device# show ipv6 ospf neighbor
Total number of neighbors in all states: 1
Number of neighbors in state Full      : 1
RouterID        Pri State    DR             BDR            Interface     [State]
2.2.2.2           1 Full     2.2.2.2        1.1.1.1        Te 1/0/2      [BDR]
```

7. The following example of the **show ipv6 ospf virtual-neighbor** command shows information about an OSPFv3 virtual neighbor.

```
device# show ipv6 ospf virtual-neighbor

Index Router ID        Address                       State     Interface
1     2.2.2.2          2001:2:0:2::2                 Full      Te 1/0/2
                Option: 00-00-00    QCount: 0    Timer: 46
```

8. The following example of the **show ipv6 ospf virtual-links** command shows information about OSPFv3 virtual links.

```
device# show ipv6 ospf virtual-links
Transit Area ID  Router ID        Interface Address             State
1                1.1.1.1          2001:2:0:2::2                 P2P
  Timer intervals(sec) :
    Hello 10, Hello Jitter 10,  Dead 40, Retransmit 5, TransmitDelay 1
  DelayedLSAck:     4 times
  Authentication: Not Configured
  Statistics:
      Type     tx         rx         tx-byte    rx-byte
      Unknown  0          0          0          0
      Hello    79         77         3156       3076
      DbDesc   4          5          352        500
      LSReq    2          2          200        152
      LSUpdate 9          10         1056       1188
      LSAck    5          5          460        440
      OSPF messages dropped,no authentication: 0
  Neighbor: State: Full Address: 2001:1:0:2::1 Interface: Te 2/0/2
```

9. The following example of the **show ipv6 ospf database** command with the **type-7** keyword shows detailed output for a configured NSSA.

```
device# show ipv6 ospf database type-7

Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
4               Typ7 1          5.5.5.5         80000001 280  58ee  60    Yes
     Bits: EF-
     Metric: 0
     Prefix Options: P,
     Referenced LSType: 0
     Prefix: 100::100/128
     Forwarding-Address: 2001:5:0:3::5
Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
4               Typ7 2          3.3.3.3         80000002 8    02c9  44    Yes
     Bits: EF-
     Metric: 33
     Prefix Options:
     Referenced LSType: 0
     Prefix: ::/0
     Forwarding-Address: 2001:3:0:2::3
```

10. The following example of the **show ipv6 ospf routes** command shows output for OSPFv3 routes.

```
device# show ipv6 ospf routes

Destination                    Cost       E2Cost     Tag        Flags    Dis
E2 ::/0                           1          33         0          00000027 110
Next_Hop_Router                Outgoing_Interface Adv_Router
2001:3:0:2::3                  Te 4/0/2           3.3.3.3
Destination                    Cost       E2Cost     Tag        Flags    Dis
E2 100::100/128                   1          0          0          0000000b 110
Next_Hop_Router                Outgoing_Interface Adv_Router
2001:5:0:3::5                  Te 4/0/3           5.5.5.5
```

11. The following example of the **show ipv6 ospf database as-external** command shows information about external LSAs.

```
device# show ipv6 ospf database as-external

Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
N/A             Extn 106        6.6.6.6         80000001 533  7993  36    Yes
     Bits: E--
     Metric: 0
     Prefix Options:
     Referenced LSType: 0
     Prefix: 4001::/64

Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
N/A             Extn 105        6.6.6.6         80000001 533  d230  44    Yes
     Bits: E--
     Metric: 0
     Prefix Options:
     Referenced LSType: 0
     Prefix: 2::2/128
```

12. The following example of the **show ipv6 ospf database** command shows information about different OSPFv3 LSAs.

```
device# show ipv6 ospf database
LSA Key - Rtr:Router Net:Network Inap:InterPrefix Inar:InterRouter
          Extn:ASExternal Grp:GroupMembership Typ7:Type7 Link:Link
          Iap:IntraPrefix Grc:Grace

Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
0               Link 145        6.6.6.6         80000002 1036 0432  56    Yes
0               Link 7          1.1.1.1         80000002 595  b6b2  56    Yes
0               Rtr  0          6.6.6.6         80000004 560  d1f4  40    Yes
0               Rtr  0          1.1.1.1         80000004 1756 ce98  40    Yes
0               Net  145        6.6.6.6         80000002 560  c899  32    Yes
0               Iap  4350       6.6.6.6         80000002 560  903a  44    Yes
1               Rtr  0          1.1.1.1         80000003 1756 0c20  24    Yes
N/A             Extn 106        6.6.6.6         80000002 560  7794  36    Yes
N/A             Extn 108        6.6.6.6         80000001 1678 9888  36    Yes
```

13. The following example of the **show ipv6 ospf database** command with the **tree** shows information about the SPF trees.

```
device# show ipv6 ospf spf tree
 SPF tree for Area 0
 +- 1.1.1.1 cost 0
     +- 6.6.6.6:145 cost 1
         +- 6.6.6.6:0 cost 1

 SPF tree for Area 1
 +- 1.1.1.1 cost 0
```

14. The following example of the **show ipv6 ospf database** command with the **table** shows information about the SPF table.

```
device# show ipv6 ospf spf table
SPF table for Area 4
  Destination           Bits Options  Cost  Nexthop                    Interface
R 3.3.3.3              ---EB V6---NR--    1  fe80::227:f8ff:feca:bbb8  Te 4/0/2
R 5.5.5.5              ---E- V6---NR--    1  fe80::205:33ff:fee5:4eda  Te 4/0/3
N 4.4.4.4[2]           ----- V6---NR--    1  ::                        Te 4/0/2
N 5.5.5.5[3]           ----- V6---NR--    1  ::                        Te 4/0/3
```

15. The following example of the **show ipv6 ospf redistribute route** command shows information about routes that the device has redistributed into OSPFv3.

```
device# show ipv6 ospf redistribute route

Id     Prefix                                    Protocol  Metric Type  Metric
105    2::2/128                                  Connect   Type-2       0
108    2001:6:4::/64                             Connect   Type-2       0
106    4001::/64                                 Connect   Type-2       0
```

16. The following example of the **show ipv6 ospf routes** command shows information about a specified OSPFv3 route.

```
device# show ipv6 ospf routes 2001:3:0:1::/64
Destination                    Cost       E2Cost     Tag        Flags    Dis
OA 2001:3:0:1::/64               2          0          0          00000003 110
Next_Hop_Router                Outgoing_Interface Adv_Router
fe80::227:f8ff:feca:bbb8       Te 4/0/2           3.3.3.3
```

Displaying OSPFv3 results

The **show ipv6 ospf** command and its variations can be used to display information about OSPFv3 configurations.

Use one or more of the following commands to verify OSPFv3 information. Using the **show ipv6 ospf** command is optional, and the variations of the command can be entered in any order.

1. Enter the **show ipv6 ospf** command to display general OSPFv3 information.

```
device# show ipv6 ospf
OSPFv3 Process number 0 with Router ID 0xc0a80001(192.168.0.1)
Running 0 days 0 hours 49 minutes 2 seconds
Number of AS scoped LSAs is 1
Sum of AS scoped LSAs Checksum is 0000d72f
External LSA Limit is 250000
Database Overflow Interval is 10
Database Overflow State is NOT OVERFLOWED
Route calculation executed 10 times
Pending outgoing LSA count 0
Authentication key rollover interval 300 seconds
Number of areas in this router is 3
Router is operating as ABR
Router is operating as ASBR, Redistribute: STATIC
High Priority Message Queue Full count: 0
BFD is disabled, BFD HoldoverInterval: 0
Graceful restart helper is enabled, strict lsa checking is disabled
Nonstop Routing is enabled
Administrative Distance
- External Routes: 110
- Intra Area Routes: 110
- Inter Area Routes: 110
Maximum Paths: 8
```

The example output offers general OSPFv3 information and indicates that the device is not operating as an ASBR. If the device is not operating as an ASBR, there is no information about redistribution in the output.

2. The following example of the **show ipv6 ospf summary** command shows summary output for the one IPv6 OSPF session that is configured.

```
device# show ipv6 ospf summary
Total number of IPv6 OSPF instances: 1

Seq Instance                        Intfs   Nbrs    Nbrs-Full LSAs    Routes
1   default-vrf                     3       1       1         19      3
```

3. The following example of the **show ipv6 ospf summary all-vrfs total** command shows the cumulative summary output for OSPFv3 sessions.

```
device# show ipv6 ospf summary all-vrfs total
----------------------------------------
         IPv6 OSPF Summary Total
----------------------------------------
Number of instances: 1024
Number of interfaces: 2048
Number of neighbors: 2048
Number of neighbors  in FULL state: 2048
Number of LSAs: 76786
Number of Routes: 67570
```

4. The following example of the **show ipv6 ospf neighbor** command shows OSPFv3 neighbor information for the device.

   NOTE
   QCount is the LSA retransmission list count. The number displays the number of LSAs that are waiting for acknowledgements.

```
device# show ipv6 ospf neighbor
Total number of neighbors in all states: 1
Number of neighbors in state Full      : 1

RouterID        Pri State    DR              BDR             Interface     State    QCount
192.168.0.2       1 Full     192.168.0.1     192.168.0.2     Eth 0/1       DR       0
```

5. The following example of the **show ipv6 ospf neighbor detail** command shows detailed OSPFv3 neighbor information for the device.

```
device# show ipv6 ospf neighbor detail
Total number of neighbors in all states: 1
Number of neighbors in state Full      : 1

RouterID        Pri State    DR              BDR             Interface     State    QCount
192.168.0.2       1 Full     192.168.0.1     192.168.0.2     Eth 0/1       DR       0
                Option: 00-00-00    Timer: 683
BFD State: NONE, BFD HoldoverInterval(sec):Configured: 0 Current: 0
```

6. The following example of the **show ipv6 ospf memory** command shows memory allocation information for OSPFv3.

```
device# show ipv6 ospf memory vrf vrf_1
   Total Dynamic Memory Allocated for this instance : 87046288 bytes
global shared memory pool for all instances
   Memory Type               Size       Allocated  Max-alloc  Alloc-Fails
   MTYPE_OSPF6_AREA          1100       1024       1065       0
   MTYPE_OSPF6_AREA_RANGE    52         0          16         0
   MTYPE_OSPF6_SUMMARY_ADDRE 36         0          16         0
   MTYPE_OSPF6_IF            396        2048       2625       0
   MTYPE_OSPF6_NEIGHBOR      24916      2048       2098       0
   MTYPE_OSPF6_ROUTE_NODE    36         71666      72415      0
   MTYPE_OSPF6_ROUTE_INFO    52         71666      80537      0
   MTYPE_OSPF6_PREFIX        24         0          16         0
   MTYPE_OSPF6_LSA           252        76787      133135     0
   MTYPE_OSPF6_VERTEX        196        5120       5327       0
   MTYPE_OSPF6_SPFTREE       60         1024       1056       0
   MTYPE_OSPF6_NEXTHOP       32         5134       8192       0
   MTYPE_OSPF6_EXTERNAL_INFO 52         0          1024       0
   MTYPE_THREAD              68         14703      15192      0
   MTYPE_OSPF6_LINK_LIST     44         6849444    7050698    0
   MTYPE_OSPF6_LINK_NODE     28         170996     265654     0
   MTYPE_OSPF6_LSA_RETRANSMI 20         0          25598      0
Global Memory Pool Usage for all instances : 415468328 bytes
global Heap memory for all instances
   Memory Type               Size       Allocated  Max-alloc  Alloc-Fails
   MTYPE_OSPF6_TOP           41104      1024       1024       0
   MTYPE_OSPF6_LSA_HDR       416        76787      107723     0
   MTYPE_OSPF6_RMAP_COMPILED 0          0          0          0
   MTYPE_OSPF6_OTHER         96         132591     137421     0
   MTYPE_THREAD_MASTER       200        1024       1024       0
---------------------
Packet Tx thread Info
---------------------
Queue Id[0]: Enqueued[291763] Dequeued [291763]
Queue Id[1]: Enqueued[13108] Dequeued [13108]
Send Failed Packets - 0
device#
```

7. The following example of the **show ipv6 ospf area** command shows detailed output for assigned OSPFv3 Area 0.

```
device# show ipv6 ospf area 0
Area 0:
  Authentication: Not Configured
  Active interface(s)attached to this area: Eth 0/1
  Inactive interface(s)attached to this area: None
  Number of Area scoped LSAs is 6
  Sum of Area LSAs Checksum is 000370d3
  Statistics of Area 0:
    SPF algorithm executed 7 times
    SPF last updated: 145 sec ago
    Current SPF node count: 3
      Router: 2 Network: 1
      Maximum of Hop count to nodes: 2
```

8. The following example of the **show ipv6 ospf interface brief** command shows limited OSPFv3 interface information.

```
device# show ipv6 ospf interface brief
Interface     Area            Status Type Cost  State    Nbrs(F/C)
Eth 0/1       0               up     BCST 1     DR       1/1
Lo 1          1               up     BCST 1     Loopback 0/0
Lo 2          2               up     BCST 1     Loopback 0/0
```

9. The following example of the **show ipv6 ospf virtual-neighbor** command shows information about an OSPFv3 virtual neighbor.

```
device# show ipv6 ospf virtual-neighbor
Index Router ID        Address                      State     Interface
1     192.168.0.2      2018:6::5                    Full      Eth 0/1
                Option: 00-00-00    QCount: 0    Timer: 578
```

10. The following example of the **show ipv6 ospf virtual-links** command shows information about OSPFv3 virtual links.

```
device# show ipv6 ospf virtual-links
Transit Area ID  Router ID        Interface Address             State
1                192.168.0.2      2018:6::4                     P2P
  Timer intervals(sec) :
    Hello 10, Hello Jitter 10,  Dead 40, Retransmit 5, TransmitDelay 1
  DelayedLSAck:     3 times
  Authentication: Not Configured
  Statistics:
      Type     tx         rx         tx-byte    rx-byte
      Unknown  0          0          0          0
      Hello    9          9          356        360
      DbDesc   3          2          104        136
      LSReq    1          1          64         28
      LSUpdate 8          7          600        784
      LSAck    7          5          312        240
      OSPF messages dropped,no authentication: 0
  Neighbor: State: Full Address: 2018:6::5 Interface: Eth 0/1
```

11. The following example of the **show ipv6 ospf database** command with the **type-7** keyword shows detailed output for a configured NSSA.

```
device# show ipv6 ospf database type-7
LSA Key - Rtr:Router Net:Network Inap:InterPrefix Inar:InterRouter
          Extn:ASExternal Grp:GroupMembership Typ7:Type7 Link:Link
          Iap:IntraPrefix Grc:Grace


Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
2               Typ7 1          192.168.0.1     80000002 211  f5fd  44    Yes
     Bits: EF-
     Metric: 10
     Prefix Options:
     Referenced LSType: 0
     Prefix: ::/0
     Forwarding-Address: 4321::56
2               Typ7 2          192.168.0.1     80000001 143  a5b2  60    Yes
     Bits: EF-
     Metric: 1
     Prefix Options:
     Referenced LSType: 0
     Prefix: 9876::5/128
     Forwarding-Address: 4321::56
```

12. The following example of the **show ipv6 ospf routes** command shows output for OSPFv3 routes.

```
device# show ipv6 ospf routes
  Current Route count: 3
    Intra: 3 Inter: 0 External: 0 (Type1 0/Type2 0)
    Equal-cost multi-path: 0
    OSPF Type: IA- Intra, OA - Inter, E1 - External Type1, E2 - External Type2
   Destination                    Cost       E2Cost     Tag        Flags    Dis
IA 1234::56/128                   0          0          0          00000002 110
   Next_Hop_Router                Outgoing_Interface Adv_Router
   ::                             Lo 1               192.168.0.1
   Destination                    Cost       E2Cost     Tag        Flags    Dis
IA 2018:6::/64                    1          0          0          00000003 110
   Next_Hop_Router                Outgoing_Interface Adv_Router
   ::                             Eth 0/1            192.168.0.1
   Destination                    Cost       E2Cost     Tag        Flags    Dis
IA 4321::56/128                   0          0          0          00000002 110
   Next_Hop_Router                Outgoing_Interface Adv_Router
   ::                             Lo 2               192.168.0.1
```

13. The following example of the **show ipv6 ospf database as-external** command shows information about external LSAs.

```
device# show ipv6 ospf database as-external
LSA Key - Rtr:Router Net:Network Inap:InterPrefix Inar:InterRouter
          Extn:ASExternal Grp:GroupMembership Typ7:Type7 Link:Link
          Iap:IntraPrefix Grc:Grace


Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
N/A             Extn 2          192.168.0.1     80000001 221  d72f  44    Yes
     Bits: E--
     Metric: 1
     Prefix Options:
     Referenced LSType: 0
     Prefix: 9876::5/128
```

14. The following example of the **show ipv6 ospf database** command shows information about different OSPFv3 LSAs.

```
device# show ipv6 ospf database
LSA Key - Rtr:Router Net:Network Inap:InterPrefix Inar:InterRouter
          Extn:ASExternal Grp:GroupMembership Typ7:Type7 Link:Link
          Iap:IntraPrefix Grc:Grace


Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
0               Link 2          192.168.0.1     80000001 790  8399  56    Yes
0               Link 2          192.168.0.2     80000001 652  e939  56    Yes
0               Rtr  0          192.168.0.1     80000003 309  2b16  40    Yes
0               Rtr  0          192.168.0.2     80000002 649  1e26  40    Yes
0               Net  2          192.168.0.1     80000001 648  eafb  32    Yes
0               Inap 1          192.168.0.1     80000001 790  2d79  44    Yes
0               Inap 10         192.168.0.1     80000001 309  0b74  44    Yes
0               Iap  60         192.168.0.1     80000001 648  96dd  44    Yes
1               Rtr  0          192.168.0.1     80000002 309  23a0  24    Yes
1               Inap 4          192.168.0.1     80000001 790  1c2d  36    Yes
1               Inap 11         192.168.0.1     80000001 309  017d  44    Yes
1               Iap  0          192.168.0.1     80000002 1233 e227  52    Yes
2               Rtr  0          192.168.0.1     80000001 309  4975  24    Yes
2               Inap 7          192.168.0.1     80000001 309  f0af  44    Yes
2               Inap 9          192.168.0.1     80000001 309  e95a  36    Yes
2               Typ7 1          192.168.0.1     80000002 309  f5fd  44    Yes
2               Typ7 2          192.168.0.1     80000001 241  a5b2  60    Yes
2               Iap  0          192.168.0.1     80000001 309  0ede  52    Yes
N/A             Extn 2          192.168.0.1     80000001 241  d72f  44    Yes
```

15. The following example of the **show ipv6 ospf spf tree** command with the **tree** shows information about the SPF trees.

```
device# show ipv6 ospf spf tree
SPF tree for Area 0
+- 192.168.0.1 cost 0
     +- 192.168.0.1:2(N) cost 1
         +- 192.168.0.2:0 cost 1

SPF tree for Area 1
+- 192.168.0.1 cost 0

SPF tree for Area 2
+- 192.168.0.1 cost 0
```

16. The following example of the **show ipv6 ospf spf table** command with the **table** shows information about the SPF table.

```
device# show ipv6 ospf spf table
SPF table for Area 0
  Destination           Bits Options  Cost  Nexthop                   Interface
R 192.168.0.2          ----- V6E---R--    1  fe80::629c:9fff:fe5a:8919 Eth 0/1
N 192.168.0.1[2]       ----- V6E---R--    1  ::                         Eth 0/1

SPF table for Area 1
  Destination           Bits Options  Cost  Nexthop                   Interface

SPF table for Area 2
  Destination           Bits Options  Cost  Nexthop                   Interface
```

17. The following example of the **show ipv6 ospf redistribute route** command shows information about routes that the device has redistributed into OSPFv3.

```
device# show ipv6 ospf redistribute route
Id     Prefix                                    Protocol  Metric Type  Metric
2      9876::5/128                               Static    Type-2       1
```

18. The following example of the **show ipv6 ospf routes** command shows information about a specified OSPFv3 route.

```
device# show ipv6 ospf routes 1234::56/128
   Destination                   Cost       E2Cost     Tag        Flags    Dis
IA 1234::56/128                  0          0          0          00000002 110
   Next_Hop_Router               Outgoing_Interface Adv_Router
   ::                            Lo 1               192.168.0.1
```

Supported scale for OSPFv3 and performance considerations

Scalability of OSPFv3 gets limited by availability of resources like Memory, CPU and hardware tables. While increasing the scale, you need to consider time taken for adjacency formation and route convergence.

The supported scale for OSPFv3 across all VRF instances is as follows:

TABLE 42 Supported scale across all VRFs

Description	Number supported
Maximum number of OSPFv3 VRF Instances	1024
Maximum number of OSPFv3 Areas	1024
Number of OSPFv3 Interfaces	2048
Number of OSPFv3 Neighbors	2048
Maximum Number of OSPFv3 Routes	64k (65536)

The supported scale for OSPFv3 per VRF instance is as follows:

TABLE 43 Supported scale per VRF

Description	Number supported
Maximum number of OSPFv3 Areas	10
Number of OSPFv3 Interfaces	256
Number of OSPFv3 Neighbors	256
Maximum Number of OSPFv3 Routes	64k (65536)

Displaying OSPFv3 results

The **show ipv6 ospf** command and its variations can be used to display information about OSPFv3 configurations.

Use one or more of the following commands to verify OSPFv3 information. Using the **show ipv6 ospf** command is optional, and the variations of the command can be entered in any order.

1. Enter the **show ipv6 ospf** command to display general OSPFv3 information.

```
device# show ipv6 ospf

 OSPFv3 Process number 0 with Router ID 0x01010101(1.1.1.1)
 Running 0 days 0 hours 0 minutes 40 seconds
 Number of AS scoped LSAs is 0
 Sum of AS scoped LSAs Checksum is 00000000
 External LSA Limit is 250000
 Database Overflow Interval is 10
 Database Overflow State is NOT OVERFLOWED
 Route calculation executed 3 times
 Pending outgoing LSA count 2
 Authentication key rollover interval 300 seconds
 Number of areas in this router is 1
 High Priority Message Queue Full count: 0
 BFD is disabled, BFD HoldoverInterval: 0
 Graceful restart helper is enabled, strict lsa checking is disabled
 Nonstop Routing is enabled
 Administrative Distance
 - External Routes: 110
 - Intra Area Routes: 110
 - Inter Area Routes: 110
 Maximum Paths: 8
```

   The example output offers general OSPFv3 information and indicates that the device is not operating as an ASBR. If the device is not operating as an ASBR, there is no information about redistribution in the output.

2. The following example of the **show ipv6 ospf summary** command shows summary output for the one IPv6 OSPF session that is configured.

```
device# show ipv6 ospf summary

Total number of IPv6 OSPF instances: 1

Seq Instance                        Intfs   Nbrs    Nbrs-Full LSAs    Routes
1   default-vrf                     1       1       1         6       1
```

3. The following example of the **show ipv6 ospf area** command shows detailed output for assigned OSPFv3 Area 0.

```
device# show ipv6 ospf area
Area 0:
  Authentication: Not Configured
  Active interface(s)attached to this area: Ve 200
  Inactive interface(s)attached to this area: None
  Number of Area scoped LSAs is 4
  Sum of Area LSAs Checksum is 0004a7a5
  Statistics of Area 0:
    SPF algorithm executed 4 times
    SPF last updated: 76 sec ago
    Current SPF node count: 3
      Router: 2 Network: 1
      Maximum of Hop count to nodes: 2
```

4. The following example of the **show ipv6 ospf interface brief** command shows limited OSPFv3 interface information.

```
device# show ipv6 ospf interface brief

Interface     Area            Status Type Cost  State    Nbrs(F/C)
Ve 200        0               up     BCST 1     BDR      1/1
```

5. The following example of the **show ipv6 ospf neighbor** command shows OSPFv3 neighbor information for the device.

```
device# show ipv6 ospf neighbor

Total number of neighbors in all states: 1
Number of neighbors in state Full      : 1

RouterID        Pri State    DR              BDR             Interface     [State]
2.2.2.2           1 Full     2.2.2.2         1.1.1.1         Ve 200        [BDR]
```

6. The following example of the **show ipv6 ospf routes** command shows output for OSPFv3 routes.

```
device# show ipv6 ospf routes

  Current Route count: 1
    Intra: 1 Inter: 0 External: 0 (Type1 0/Type2 0)
    Equal-cost multi-path: 0
    OSPF Type: IA- Intra, OA - Inter, E1 - External Type1, E2 - External Type2
   Destination                    Cost       E2Cost     Tag        Flags    Dis
IA 2001::/64                      1          0          0          00000003 110
   Next_Hop_Router                Outgoing_Interface Adv_Router
   ::                             Ve 200             2.2.2.2
```

7. The following example of the **show ipv6 ospf database** command shows information about different OSPFv3 LSAs.

```
device# show ipv6 ospf database

LSA Key - Rtr:Router Net:Network Inap:InterPrefix Inar:InterRouter
          Extn:ASExternal Grp:GroupMembership Typ7:Type7 Link:Link
          Iap:IntraPrefix Grc:Grace


Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
0               Link 6472       1.1.1.1         80000001 371  93e1  56    Yes
0               Link 6416       2.2.2.2         80000001 382  d73d  56    Yes
0               Rtr  0          2.2.2.2         80000003 341  bdfc  40    Yes
0               Rtr  0          1.1.1.1         80000002 341  f096  40    Yes
0               Net  6416       2.2.2.2         80000001 341  b536  32    Yes
0               Iap  192480     2.2.2.2         80000001 341  d8bf  44    Yes
```

8. The following example of the **show ipv6 ospf spf tree** command with the **tree** shows information about the SPF trees.

```
device# show ipv6 ospf spf tree

 SPF tree for Area 0
 +- 1.1.1.1 cost 0
     +- 2.2.2.2:6416(N) cost 1
         +- 2.2.2.2:0 cost 1
```

9. The following example of the **show ipv6 ospf spf table** command with the **table** shows information about the SPF table.

```
device# show ipv6 ospf spf table

 SPF table for Area 0
  Destination           Bits Options  Cost  Nexthop                    Interface
R 2.2.2.2              ----- V6E---R--    1  fe80::629c:9fff:fe5b:801  Ve 200
N 2.2.2.2[6416]        ----- V6E---R--    1  ::                         Ve 200
```

10. The following example of the **show ipv6 ospf routes** command shows information about a specified OSPFv3 route.

```
device# show ipv6 ospf routes  2001::/64

   Destination                    Cost       E2Cost     Tag        Flags    Dis
IA 2001::/64                      1          0          0          00000003 110
   Next_Hop_Router                Outgoing_Interface Adv_Router
   ::                             Ve 200             2.2.2.2
```

Clearing OSPFv3 redistributed routes

OSPFv3 redistributed routes for a device can be cleared using a CLI command.

The **show ipv6 ospf redistribute route** command is entered to verify that there are IPv6 routes on the device that have been redistributed into OSPFv3. The **clear ipv6 ospf redistribution** command is entered to clear all OSPFv3 redistributed routes. The **show ipv6 ospf redistribute route** command is re-entered to verify that the OSPFv3 redistributed routes have been cleared.

1. Enter the **exit** command. Repeat as necessary.

```
device(config)# exit
```

   Enters privileged EXEC mode.

2. Enter the **show ipv6 ospf redistribute route** command.

```
device# show ipv6 ospf redistribute route
Id     Prefix                                  Protocol  Metric Type  Metric
34     1900:53:131::130/124                    Connect   Type-2       0
36     2001:2031::/64                          Connect   Type-2       0
37     2001:3031::/64                          Connect   Type-2       0
38     2001:3032::/64                          Connect   Type-2       0
39     2001:3033::/64                          Connect   Type-2       0
32     2001:a131:131:132::/64                  Static    Type-2       1
35     2131:131::/64                           Connect   Type-2       0
33     3001::132/128                           Connect   Type-2       0
device#
```

   Displays all IPv6 routes that the device has redistributed into OSPFv3.

3. Enter the **clear ipv6 ospf redistribution** command.

```
device# clear ipv6 ospf redistribution
```

   Clears OSPFv3 redistributed routes.

4. Enter the **show ipv6 ospf redistribute route** command.

```
device# show ipv6 ospf redistribute route
Id     Prefix                                    Protocol  Metric Type  Metric
41     1900:53:131::130/124                      Connect   Type-2       0
43     2001:2031::/64                            Connect   Type-2       0
44     2001:3031::/64                            Connect   Type-2       0
45     2001:3032::/64                            Connect   Type-2       0
46     2001:3033::/64                            Connect   Type-2       0
47     2001:a131:131:132::/64                    Static    Type-2       1
42     2131:131::/64                             Connect   Type-2       0
40     3001::132/128                             Connect   Type-2       0
device#
```

Displays all IPv6 routes that the device has redistributed into OSPFv3.

The following example clears OSPFv3 redistributed routes for a device and verifies that the OSPFv3 redistributed routes have been cleared:

```
device(config-ipv6-router-ospf-vrf)# exit
device(config-rbridge-id-122)# exit
device(config)# exit
device# show ipv6 ospf redistribute route
device# clear ipv6 ospf redistribution
device# show ipv6 ospf redistribute route
```

Multi-VRF

- Multi-VRF overview 599
- Configuring Multi-VRF 601
- Multi-VRF configuration example 605
- Inter-VRF route leaking 612
- Internet route scaling 620

Multi-VRF overview

Virtual Routing and Forwarding (VRF) allows routers to maintain multiple routing tables and forwarding tables on the same router. A Multi-VRF router can run multiple instances of routing protocols with a neighboring router with overlapping address spaces configured on different VRF instances.

Some vendors also use the terms Multi-VRF CE or VRF-Lite for this technology. VRF-Lite provides a reliable mechanism for a network administrator to maintain multiple virtual routers on the same device. The goal of providing isolation among different VPN instances is accomplished without the overhead of heavyweight protocols (such as MPLS) used in secure VPN technologies. Overlapping address spaces can be maintained among the different VPN instances.

Some vendors also use the terms Multi-VRF CE or VRF-Lite for this technology. VRF-Lite provides a reliable mechanism for a network administrator to maintain multiple virtual routers on the same device. The goal of providing isolation among different VPN instances is accomplished without the overhead of heavyweight protocols used in secure VPN technologies. Overlapping address spaces can be maintained among the different VPN instances.

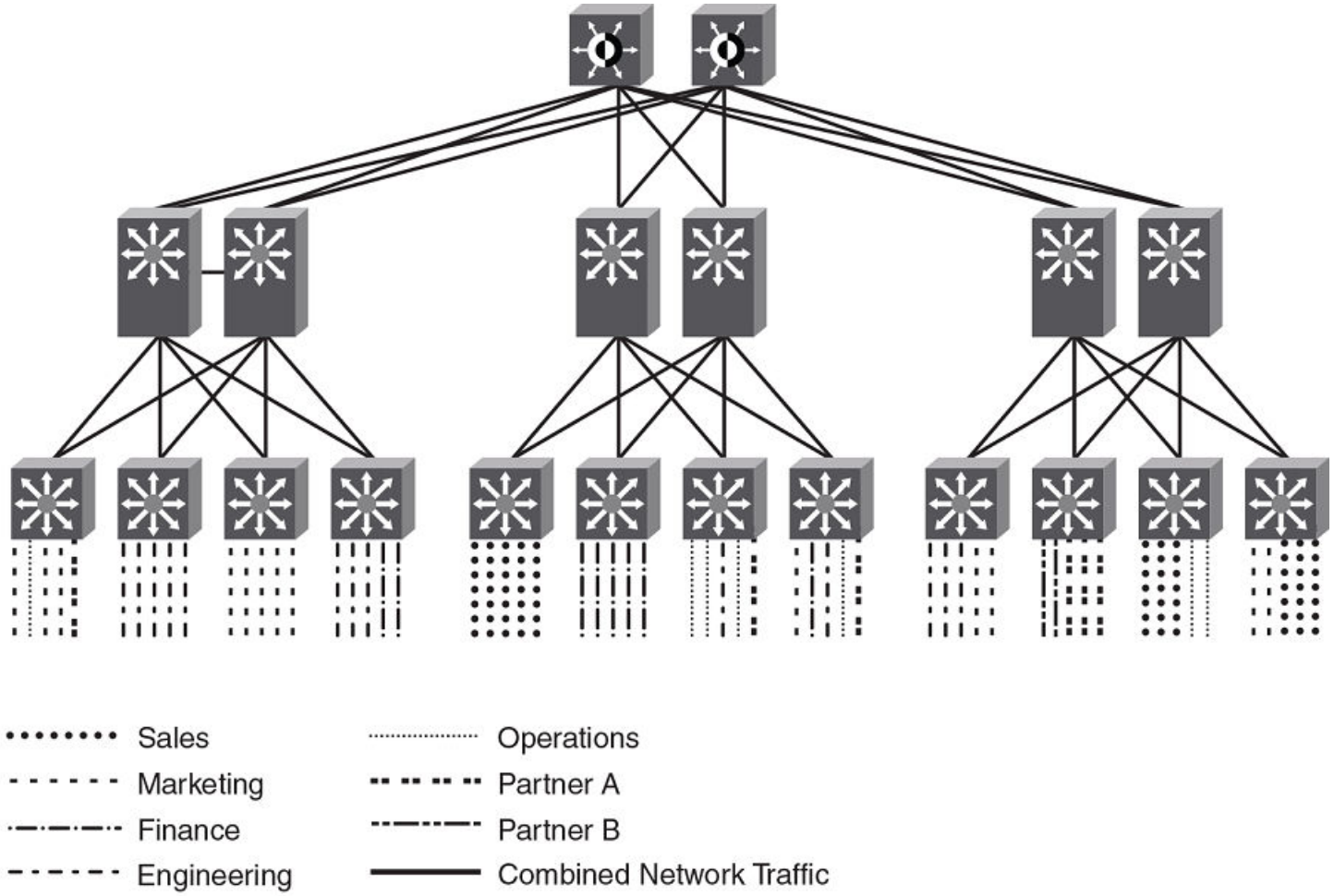
Central to VRF-Lite is the ability to maintain multiple VRF tables on the same Provider Edge (PE) Router. VRF-Lite uses multiple instances of a routing protocol such as OSPF or BGP to exchange route information for a VPN among peer PE routers. The VRF-Lite capable PE router maps an input customer interface to a unique VPN instance. The router maintains a different VRF table for each VPN instance on that PE router. Multiple input interfaces may also be associated with the same VRF on the router, if they connect to sites belonging to the same VPN. This input interface can be a physical interface or a virtual Ethernet interface on a port.

In Multi-VRF deployments:

- Two VRF-capable routers must be directly connected at Layer 3, deploying BGP, OSPF, or static routes.
- Each VRF maintains unique routing and forwarding tables.
- Each VRF can be assigned one or more Layer 3 interfaces on a router to be part of the VRF.
- Each VRF can be configured with IPv4 address family, IPv6 unicast address family, or both.
- A packet's VRF instance is determined based on the VRF index of the interface on which the packet is received.
- Separate routing protocol instances are required for each VRF instance.
- Overlapping address spaces can be configured on different VRF instances.

Multi-VRF deployments provide the flexibility to maintain multiple virtual routers, which are segregated for each VRF instance. The following illustrates a generic, high-level topology where different enterprise functions are assigned unique VRF instances.

FIGURE 78 Example high-level Multi-VRF topology



A Multi-VRF instance can be configured on any of the following:

- Virtual interfaces
- Loopback interfaces
- Ethernet interfaces
- Tunnel interfaces - The tunnel can belong to any user-defined VRF, but the tunnel source and tunnel destination are restricted to the default VRF.

A Multi-VRF instance **cannot** be configured on any of the following:

- Physical interfaces
- Management interfaces

To configure Multi-VRF, perform the following steps:

- (Optional) Configure tagging on peer interfaces for security.
- Configure VRF instances.
- (Optional) Configure a Route Distinguisher (RD) for new VRF instances.
- Configure an IPv4 address family or IPv6 unicast address family (AF) for new VRF instances.

- Configure routing protocols for new Multi-VRF instances.
- Assign VRF instances to Layer 3 interfaces.

  NOTE
  For the details of Multi-VRF in a BGP EVPN context, refer to "Multi-VRF for BGP EVPN" in the "BGP EVPN" chapter in this document.

In addition, VRFs operate without knowledge of one another unless they are imported or exported into one another by means of inter-VRF route leaking. For details and configuration examples, refer to "Inter-VRF route leaking" in the "VRF" chapter in the *Network OS Layer 3 Routing Configuration Guide*.

In addition, VRFs operate without knowledge of one another unless they are imported or exported into one another by means of inter-VRF route leaking. For details and configuration examples, refer to "Inter-VRF route leaking" in this chapter.

Configuring Multi-VRF

Configuring a VRF instance

Do the following to configure a VRF instance.

A device can be configured with more than one VRF instance. You should define each VRF instance before assigning the VRF to a Layer 3 interface. The range of the instance name is from 1 through 255 alphanumeric characters. An optional router ID can also be assigned.

Use the **address-family** command in VRF configuration mode to specify an IPv4 or IPv6 address family. For a specific address family you can also configure static route, static ARP, and static route, IPv6 neighbor, and multicast for IPv6.

ATTENTION
Using the **overwrite** option while downloading a configuration from a TFTP server to the running-config will lead to the loss of all VRF configurations when a VRF is configured on a routing interface.

1. Enter global configuration mode and create a VRF instance.

```
device# configure terminal
device(config)# vrf corporate
device(config-vrf-corporate)#
```

2. (Optional) Assign a router ID.

```
device(config-vrf-corporate)# ip router-id 1.1.1.1
```

3. Use the **address-family unicast (VRF)** command to configure an address family on the VRF and exit. This example uses IPv4.

```
device(config-vrf-corporate)# address-family ipv4 unicast
device(config-vrf-corporate-ipv4)# exit
```

4. Verify the configuration.

```
device(config-vrf-corporate)# do show vrf
Total number of VRFs configured: 4
VrfName                          VrfId  V4-Ucast  V6-Ucast
corporate                        3      Enabled   Disabled
default-vrf                      1      Enabled   Enabled
mgmt-vrf                         0      Enabled   Enabled
test1                            2      Enabled   Enabled
```

Starting a routing process for a VRF

You must enable a routing protocol for each VRF instance. This example uses OSPF.

1. In global configuration mode, enable OSPF for the VRF instance "corporate."

```
device(config)# router ospf vrf corporate
```

2. Configure the VRF to use OSPF Area 0.

```
device(config-ospf-router-vrf-corporate)# area 0
```

3. (Optional) Configure the VRF to ensure that essential OSPF neighbor state changes are logged, especially in the case of errors.

```
device(config-ospf-router-vrf-corporate)# log adjacency
```

Assigning a Layer 3 interface to a VRF

The following example illustrates how a virtual Ethernet (VE) interface is assigned to a VRF, and how IP addresses and the OSPF protocol are configured.

> **ATTENTION**
> After a VRF instance is configured on the device, one or more Layer 3 interfaces (physical or virtual Ethernet) must be assigned to the VRF. This causes all existing IP addresses to be deleted; this action also triggers cache deletion, route deletion, and associated cleanup. After assigning an interface to the VRF, the user must reconfigure the IP address and interface properties.

1. Enter global configuration mode.

```
device(config)# configure terminal
```

2. Enter the **interface ve** command to specify a virtual Ethernet (VE) interface and enter VE configuration mode.

```
device(config)# interface ve 10
```

3. In VE configuration mode, enable forwarding for the VRF "guest".

```
device(config-if-Ve-10)# vrf forwarding guest
```

4. Configure an IPv4 address and mask on the VE interface.

```
device(config-if-Ve-10)# ip address 192.168.1.254/24
```

5. Enable OSPF Area 0.

```
device(config-if-Ve-10)# ip ospf area 0
```

6. Configure the interface as passive.

```
device(config-if-Ve-10)# ip ospf passive
```

7. Exit the configuration.

```
device(config-if-Ve-10)# exit
```

Assigning a loopback interface to a VRF

Do the following to assign a loopback interface to a nondefault VRF.

Because a loopback interface is always available as long as the device is available, it allows routing protocol sessions to stay up even if the outbound interface is down. Assigning a loopback interface to a VRF is similar to assigning any interface. A loopback interface that is not assigned to a nondefault VRF belongs to the default VRF.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **interface loopback** command to specify a loopback interface and enter interface loopback configuration mode.

```
device(config)# interface loopback 1
device(config-Loopback-1)#
```

3. Use the **vrf forwarding** command to assign the interface to the VRF "customer-1" in this example.

```
device(config-Loopback-1)# vrf forwarding customer-1
```

4. Assign an IPv4 address and mask to the loopback interface.

```
device(config-Loopback-1)# ip address 10.0.0.1/32
```

Verifying a Multi-VRF configuration

The following examples illustrate the use of a variety of show commands that are useful in verifying Multi-VRF configurations.

To verify all configured VRFs in summary mode, enter the **show vrf** command, as in the following example.

```
device# show vrf
Total number of VRFs configured: 4
VrfName                          VrfId  V4-Ucast  V6-Ucast
corporate                        3      Enabled   Disabled
default-vrf                      1      Enabled   Enabled
mgmt-vrf                         0      Enabled   Enabled
test1                            2      Enabled   Enabled
```

To verify a specific VRF, enter the **show vrf** *vrf-name* command, as in the following example.

```
device# show vrf corporate
VRF-Name: corporate, VRF-Id: 3
IP Router-Id: 1.1.1.1
Interfaces:
        Lo 10
Address-family IPV4 unicast
        Max routes:-   Route count:0
   No import route-maps
   No export route-maps

Address-family IPV6 unicast
        Max routes:-   Route count:2
   No import route-maps
   No Export route-maps
```

Use the **show vrf detail** command to display information about all VRFs.

The following commands display additional information about a specific application, protocol configuration, or protocol state for both the default VRF and user-defined VRFs.

TABLE 44 Additional useful show commands

Default VRF	User-defined VRF
show ip route	**show arp vrf**
show ip ospf neighbor	**show ip route vrf** *vrf-name*
show ip bgp summary	**show ip ospf neighbor vrf** *vrf-name*
	show ip bgp summary vrf *vrf-name*

Removing a VRF configuration

The following examples illustrate a variety of ways by which you can remove a VRF configuration: deleting a VRF instance from a port, deleting an address family from a VRF, and deleting the VRF globally.

To delete a VRF instance from a specific port, use the **no** form of the **vrf forwarding** command. This removes all Layer 3 interface bindings from the VRF, and returns the interface to default VRF mode. All IP addresses and protocol configuration on this Layer 3 interface are removed.

```
device(conf-if-eth-0/1)# no vrf forwarding
```

To delete an IPv4 or IPv6 address family from a VRF instance, use the **no** form of the **address-family** command. All configuration related to the address family on all ports of the VRF are removed. Routes allocated to the address family are returned to the global pool.

```
device(config-vrf-customer1)# no address-family ipv4
```

To delete a VRF instance globally, use the **no** form of the **vrf** command. All IPv4 or IPv6 addresses are removed from all interfaces.

```
device(config)# no vrf customer1
```

Configuring the maximum number of routes

You can use the **max-route** command to specify the number of routes held in the routing table per VRF instance, for an IPv4 or IPv6 VRF address family.

If this command is not used, the maximum number of routes is 4294967295. This number does not appear in a running configuration.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Specify a VRF instance (in this example, "myvrf") and enter VRF configuration mode.

   ```
   device(config)# vrf myvrf
   device(config-vrf-myvrf)#
   ```

3. Enter the **address-family unicast** command, in this example for IPv4, and enter VRF address-family IPv4 unicast configuration mode.

   ```
   device(config-vrf-myvrf)# address-family ipv4 unicast
   ```

4. Enter the **max-route** command and specify the maximum number of routes to be held in the routing table for this VRF instance, 3600 in this example. (The range is from 1 through 4294967295.)

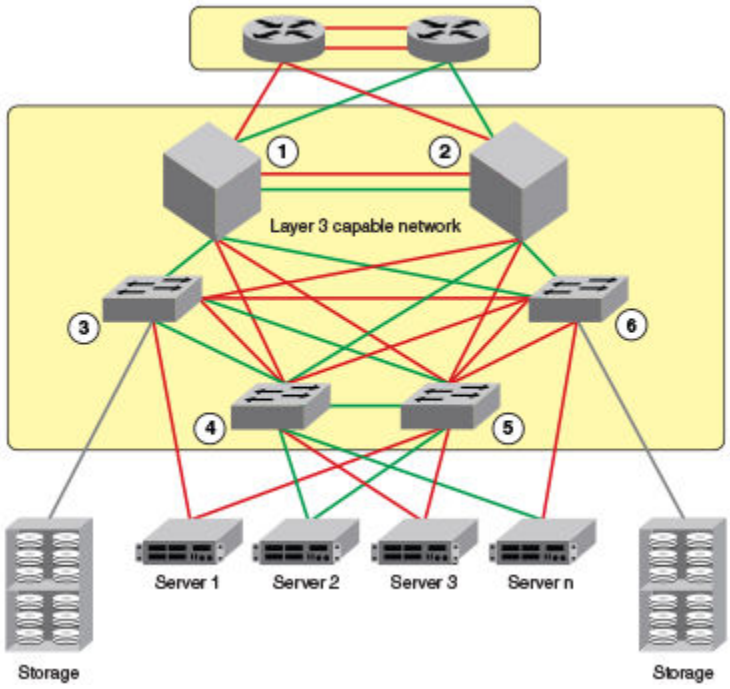
```
device(vrf-myvrf-ipv4-unicast)# max-route 3600
```

Multi-VRF configuration example

The following is an example of a basic Multi-VRF configuration that uses eBGP with OSPF.

The following example topology shows a typical network that uses the Multi-VRF feature to implement Layer 3 VPNs across two directly connected (at Layer 3) Provider Edge (PE) devices. The Customer Edge (CE) devices can be any router or Layer 3 switch that is capable of running one or many dynamic routing protocols such as BGP or OSPF, or even simple static routing. In this example, we use two devices that interconnect all four CE routers with a single link between the two of them.

FIGURE 79 eBGP configured between PE1 and PE2 with OSPF (Area 0) configured between PEs and CEs



1. PE1
2. PE2
3. CE1
4. CE2
5. CE3
6. CE4

Topology details are listed below.

TABLE 45 Topology details

Node	Description	Networks	Carries routes . . .	Interfaces
PE1	Aggregation	10.1.1.0/24 10.3.1.0/24		0/1 0/2 0/3
PE2	Aggregation	10.2.1.0/24 10.3.1.0/24		0/1 0/2 0/3
CE1	Edge	10.1.1.0/24	10.1.2.0/24 10.1.3.0/24	0/1 0/2
CE2	Edge	10.1.1.0/24	10.1.2.0/24 10.1.3.0/24	0/1 0/2
CE3	Edge	10.2.1.0/24	10.2.2.0/24 10.2.3.0/24	0/1 0/2
CE4	Edge	10.2.1.0/24	10.2.2.0/24 10.2.3.0/24	0/1 0/2

- Traffic is separated by VRF "green" on VLAN 10 and VRF "red" on VLAN 20.
- eBGP and OSPF (Area 0) are used to connect aggregation switches PE1 and PE2.
- iBGP and OSPF (Area 0) are used to connect the aggregation switches to the CE switches.
- Alternatively, with only OSPF used, Areas 1 and 2 could carry traffic between the PEs and CEs.

The following configuration examples for PE1, PE2, CE1, CE2, CE3, and CE4 implement the topology above.

> **NOTE**
> The single link between the two PEs could also be replaced by a Layer 2 switched network if direct physical connection between the PEs is not possible. The only requirement for the connections is that the two PEs be directly connected at Layer 3.

In the example topology, because two different VLANs (10 and 20) have overlapping IP address ranges, communication within each customer's VPN across the two PE routers (that is, between CE1 and CE4, and between CE2 and CE3) must be separated by means of two different VRFs ("green" and "red").

Multi-VRF with eBGP and OSPF: Configuring PE1

Two VRFs ("red" and "green") are defined. In the eBGP configuration, PE1 is defined in Local AS 1.

VRFs "green" and "red" are configured, and both have the same IP network address assigned (10.3.1.2/24). This is possible because each of the BGP VRF instances has its own BGP tables. This is also the same IP network address that will be assigned to VRFs "green" and "red" on PE2 within Local AS 2. Redistribution of OSPF routes from PE1's CE peers is enabled to all for their advertisement to PE2.

Both VRFs are configured in Area 0 and are directed to redistribute their routes to BGP. The physical interfaces to the CEs are assigned to the appropriate VRF and are configured with the same network address (10.1.1.1/24) and OSPF Area 0.

The virtual Interfaces (Ve 10 and Ve 20) are configured with the same network address (10.3.1.1/24) and for VRF forwarding in the appropriate VRF ("green" or "red").

1. In global configuration mode, create VLANs 10 and 20.

```
device(config)# vlan 10
device(config-Vlan-10)# exit
device(config)# vlan 20
```

2. From global configuration mode, enter interface subtype configuration mode, and then create a virtual Ethernet routing interface for the VLAN.

```
device(config)# vlan 10
device(config-Vlan-10)# router-interface ve 10
```

3. Repeat the above steps as appropriate for remaining physical, VLAN, and virtual Ethernet interfaces.
4. Create VRF "green".

```
device(config)# vrf green
device(config-vrf-green)# exit
```

   Repeat for every VRF instance. Use the **address-family ipv6 unicast** command for IPv6 addresses. Also, you can use the **max-route** command, which helps restrict the maximum number of routes per VRF.

5. Configure VRF "red" and exit the VRF configuration.

```
device(config)# vrf red
device(config-vrf-red)# exit
```

6. In global configuration mode, enable BGP routing and configure the following in this IPv4 example.
   a) Enable BGP routing.

```
device(config)# router bgp
```

   b) Assign a Local AS number.

```
device(config-bgp-router)# local-as 1
```

   c) Enable IPv4 unicast address-family mode for VRF "green."

```
device(config-bgp-router)# address-family ipv4 unicast vrf green
```

   d) Assign Remote AS 2 as a neighbor with the specified address.

```
device(config-bgp-ipv4u-vrf)# neighbor 10.3.1.2 remote-as 2
```

   e) Assign the appropriate network.

```
device(config-bgp-ipv4u-vrf)# network 10.3.1.0/24
```

   f) Redistribute the OSPF routes into BGP4, specifying the types of routes to be distributed, then exit the address family configuration.

```
device(config-bgp-ipv4u-vrf)# redistribute ospf match internal
device(config-bgp-ipv4u-vrf)# redistribute ospf match external1
device(config-bgp-ipv4u-vrf)# redistribute ospf match external2
device(config-bgp-ipv4u-vrf)# exit
```

7. Repeat as above for VRF "red."

```
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast vrf red
device(config-bgp-ipv4u-vrf)# neighbor 10.3.1.2 remote-as 2
device(config-bgp-ipv4u-vrf)# network 10.3.1.0/24
device(config-bgp-ipv4u-vrf)# redistribute ospf match internal
device(config-bgp-ipv4u-vrf)# redistribute ospf match external1
device(config-bgp-ipv4u-vrf)# redistribute ospf match external2
device(config-bgp-ipv4u-vrf)# exit
device(config-bgp-router)# exit
```

8. Enable OSPF routing for VRF "green" and configure the following.
   a) Enable OSPF.

```
device(config)# router ospf vrf green
```

   b) Assign Area 0.

```
device(config-ospf-router-vrf-green)# area 0
```

   c) Redistribute the OSPF routes into BGP4 and exit the VRF configuration.

```
device(config-ospf-router-vrf-green)# redistribute bgp
device(config-ospf-router-vrf-green)# exit
device(config-ospf-router)# exit
```

9. Repeat as above for VRF "red".

```
device(config)# router ospf vrf red
device(config-ospf-router-vrf-red)# area 0
device(config-ospf-router-vrf-red)# redistribute bgp
device(config-ospf-router-vrf-red)# exit
```

10. Configure the Ethernet interfaces as appropriate, as in the following example.
    a) Assign an interface to VRF instance "green" and enable forwarding.

```
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# vrf forwarding green
```

    b) Assign Area 0.

```
device(conf-if-eth-0/1)# ip ospf area 0
```

    c) Assign an IPv4 network.

```
device(conf-if-eth-0/1)# ip address 10.1.1.1/24
```

    d) Repeat as above for another Ethernet interface and VRF "red" and exit the interface configuration.

```
device(conf-if-eth-0/2)# interface ethernet 0/2
device(conf-if-eth-0/2)# vrf forwarding red
device(conf-if-eth-0/2)# ip ospf area 0
device(conf-if-eth-0/2)# ip address 10.1.1.1/24
device(conf-if-eth-0/2)# exit
```

11. Configure the VE interfaces for the appropriate VRF and network.
    a) Configure VE 10, corresponding to VLAN 10.

```
device(config)# interface ve 10
device(config-if-Ve-10)# vrf forwarding green
device(config-if-Ve-10)# ip address 10.3.1.1/24
device(config-if-Ve-10)# no shutdown
```

    b) Repeat the above for VE 20, corresponding to VLAN 20.

```
device(config-ve-10)# interface ve 20
device(config-if-Ve-20)# vrf forwarding red
device(config-if-Ve-20)# ip address 10.3.1.1/24
device(config-if-Ve-20)# no shutdown
device(config-if-Ve-20# exit
```

Multi-VRF with eBGP and OSPF: Configuring PE2

The PE2 configuration is a mirror image of the PE1 configuration. The only difference is that the BGP neighbor on the corresponding interface has an IP address of 10.3.1.1. This is used in the BGP configuration.

The following summarizes the configuration on PE2.

```
device(config)# vlan 10
device(config-vlan-10)# router-interface ve 10
device(config-vlan-10)# exit
device(config)# vlan 20
device(config-vlan-20)# router-interface ve 20
device(config-vlan-20)# exit
device(config)# vrf green
device(config-vrf-green)# exit
device(config)# vrf red
device(config-vrf-red)# exit
device(config)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# address-family ipv4 unicast vrf green
device(config-bgp-ipv4u-vrf)# neighbor 10.2.1.1 remote-as 2
device(config-bgp-ipv4u-vrf)# network 10.2.1.0/24
device(config-bgp-ipv4u-vrf)# redistribute ospf match internal
device(config-bgp-ipv4u-vrf)# redistribute ospf match external1
device(config-bgp-ipv4u-vrf)# redistribute ospf match external2
device(config-bgp-ipv4u-vrf)# exit
device(config-bgp-router)# address-family ipv4 unicast vrf red
device(config-bgp-ipv4u-vrf)# neighbor 10.3.1.1 remote-as 2
device(config-bgp-ipv4u-vrf)# network 10.3.1.0/24
device(config-bgp-ipv4u-vrf)# redistribute ospf match internal
device(config-bgp-ipv4u-vrf)# redistribute ospf match external1
device(config-bgp-ipv4u-vrf)# redistribute ospf match external2
device(config-bgp-ipv4u-vrf)# exit
device(config)# router ospf vrf green
device(config-ospf-router-vrf-green)# area 0
device(config-ospf-router-vrf-green)# redistribute bgp
device(config-ospf-router-vrf-green)# exit
device(config)# router ospf vrf red
device(config-ospf-router-vrf-red)# area 0
device(config-ospf-router-vrf-red)# redistribute bgp
device(config-ospf-router-vrf-red)# exit
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# vrf forwarding green
device(conf-if-eth-0/2)# ip ospf area 0
device(conf-if-eth-0/2)# ip address 10.2.1.1/24
device(conf-if-eth-0/2)# interface ethernet 0/3
device(conf-if-eth-0/3)# vrf forwarding red
device(conf-if-eth-0/3)# ip ospf area 0
device(conf-if-eth-0/3)# ip address 10.3.1.1/24
device(conf-if-eth-0/3)# exit
device(config)# interface ve 10
device(config-Ve-10)# vrf forwarding green
device(config-Ve-10)# ip address 10.2.1.1/24
device(config-Ve-10)# exit
device(config)# interface ve 20
device(config-ve-20)# vrf forwarding red
device(config-ve-20)# ip address 10.3.1.1/24
```

Multi-VRF with eBGP and OSPF: Configuring CE1 and CE2

The CE1 and CE2 router configurations are exactly the same. Both are configured in OSPF Area 0 with route redistribution enabled. The IP addresses 10.1.2.1/32 and 10.1.3.1/32 are configured for loopback interfaces, allowing them to carry routes from these networks.

1. Enable OSPF routing.

```
device(config)# router ospf
device(config-router-ospf-default-vrf)#
```

2. Assign Area 0.

```
device(config-router-ospf-default-vrf)#)# area 0
```

3. Redistribute connected routes into OSPF and exit the OSPF configuration.

```
device(config-router-ospf-default-vrf)# redistribute connected
device(config-router-ospf-default-vrf)# exit
device(config)#
```

4. Configure a loopback interface.

```
device(config)# interface loopback 1
device(config-Loopback-1)# ip address 10.1.1.1/32
```

5. Configure an Ethernet interface, assign it to Area 0, and assign it to the appropriate network.

```
device(config-Loopback-1)# interface ethernet 0/1
device(conf-if-eth-0/1)# ip ospf area 0
device(conf-if-eth-0/1)# ip address 10.1.2.1/24
```

6. Repeat the above for additional loopback and Ethernet interfaces.

Multi-VRF with eBGP and OSPF: Configuring CE3 and CE4

The CE3 and CE4 router configurations are exactly the same. Both are configured in OSPF Area 0 with route redistribution enabled.

The following summarizes the configuration.

```
device(config)# router ospf
device(config-router-ospf-default-vrf)# area 0
device(config-router-ospf-default-vrf)# redistribute connected
device(config-router-ospf-default-vrf)# exit
device(config)# interface loopback 1
device(config-Loopback-1)# ip address 10.1.1.1/32
device(config-Loopback-1)# exit
device(config)# interface loopback 2
device(config-Loopback-2)# ip address 10.1.1.2/32
device(config-Loopback-2)# exit
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# ip ospf area 0
device(conf-if-eth-0/1)# ip address 10.2.2.1/24
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# ip ospf area 0
device(conf-if-eth-0/2)# ip address 10.2.3.1/24
```

Inter-VRF route leaking

VRFs operate without knowledge of one another unless they are imported or exported into one another by means of inter-VRF route leaking. This feature allows the leaking of route prefixes from one VRF instance to another VRF instance on the same physical router, which eliminates the need for external routing.

Inter-VRF route leaking is useful in cases where multiple VRFs share the same path to an external domain, while maintaining the internal routing information limited to their own VRFs. This feature enables a data center to consolidate multiple VRF services onto one server.

Both static and dynamic route leaking are supported. Each routed interface (virtual or physical) can belong to only one VRF.

- Static route leaking provides a mechanism to leak manually configured route entries from a source VRF to a destination VRF.
- Dynamic route leaking provides a mechanism to leak routes learned from routing protocols such as BGP and OSPF from a source VRF to a destination VRF. You can leak routes by configuring a route map and associating this route map with a source VRF. The match criteria defined in the route map consist of specific route prefixes that exist in the source VRF.

IMPORTANT
When OptiScale™ is enabled for the SLX 9540, the non-default VRF is not supported, nor is inter-VRF route leaking. The device supports multi-VRF and inter-VRF route leaking only when OptiScale is not enabled. For more information, see Internet route scaling on page 620.

Dynamic route-leak restrictions

- Exporting of route maps is not supported.
- Match criteria in a route map must be provided with prefix lists. Other match criteria is ignored.
- Routes in the management-vrf with a next-hop as eth0 or a management interface are not leaked.

Inter-VRF route conflicts

IMPORTANT
Deploy this feature only if you are an advanced user, because route leak configuration in source VRFs can collide with route and interface definitions in target VRFs. Such a situation can lead to unpredictable behavior in packet forwarding.

Leaked route conflicts can occur in situations such as the following:

- Static route conflict
- Dynamic route conflict
- Connected route conflict

A static route conflict can occur when the same prefix is reachable by two different next hops in the target VRF. The forwarding behavior is different depending on which command occurred later. In the following example, the global configuration presents a static route conflict for 10.1.2.0/24.

```
vrf red
device(config-vrf-red)# address-family ipv4 unicast
device(config-vrf-red-ipv4-unicast)# ip route 10.1.2.0/24 next-hop-vrf green 10.1.1.1
device(config)# vrf green
device(config-vrf-green)# address-family ipv4 unicast
device(config-vrf-green)# ip route 10.1.2.0/24 18.1.1.1
```

NOTE
However, if the source of the prefix in each case is different (for example, 10.1.2.0 comes from OSPF, BGP, static, or connected, then the method of conflict resolution (where the most recent configuration takes precedence) does not apply. The order of preference is as follows: (1) connected, (2) static, (3) OSPF, and (4) BGP, assuming that the administrative distances for the prefixes are the defaults. In other words, when the prefix is installed through different sources (OSPF, BGP, static, or connected), the prefix with the lowest administrative distance takes precedence. The most recently configured prefix rule applies only if the source of the prefix is the same.

IMPORTANT
Ensure that identical prefixes are not leaked.

A dynamic route conflict can occur when dynamic routing protocols advertise different routes to the same prefix in the target VRF.

The following example illustrates a connected route conflict:

```
device(config)# vrf red
device(config-vrf-red)# address-family ipv4 unicast
device(vrf-red-ipv4-unicast)# ip route 10.1.2.0/24 next-hop-vrf green 10.1.1.1
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# vrf forwarding green
device(conf-if-eth-0/1)# ip address 10.1.2.1/24
```

NOTE
Be aware of such possible conflicts before deploying the route leak feature, because there is no method for error checking these scenarios. A best practice is to ensure that definitions are globally unique and route collisions do not exist.

Displaying inter-VRF route leaking

The **show ip route** command displays "%vrf" in the route type, with the next-hop address, for the leaked routes in a VRF. The following example displays output for routes in the default-vrf

```
device# show ip route
IP Routing Table for VRF "default-vrf"
Total number of IP routes: 6
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

1.1.1.0/24,
    *via 8.8.8.100%vrf200, Ve200 , [1/1], 2m45s, static, tag 0
2.2.2.0/24, attached
    *via DIRECT, Ve 100, [0/0], 2m45s, direct, tag 0
2.2.2.100/32, attached
    *via DIRECT, Ve 100, [0/0], 2m45s, local, tag 0
40.0.0.0/24, attached
    *via DIRECT, Eth 0/2, [0/0], 31m15s, direct, tag 0
40.0.0.1/32, attached
    *via DIRECT, Eth 0/2, [0/0], 31m15s, local, tag 0
90.0.0.0/24,
    *via 40.0.0.100, Eth 0/2, [1/1], 16m42s, static, tag 0
```

You can also determine the leaked route for a specific VRF, by using the **show ip route vrf** command, as illustrated in the following example for vrf1 that displays "%default-vrf" in the route type.

```
device# show ip route vrf vrf1
IP Routing Table for VRF "vrf1"
Total number of IP routes: 6
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

3.3.3.0/24,
    *via 14.14.14.100, Ve 2, [1/1], 0m23s, static, tag 0
```

```
11.11.11.0/24,
    *via 40.0.0.2%default-vrf, Eth 0/2, [1/1], 1m40s, static, tag 0
14.14.14.0/24, attached
    *via DIRECT, Ve 2, [0/0], 9m11s, direct, tag 0
14.14.14.14/32, attached
    *via DIRECT, Ve 2, [0/0], 9m11s, local, tag 0
15.15.15.0/24, attached
    *via DIRECT, Ve 3, [0/0], 9m11s, direct, tag 0
15.15.15.1/32, attached
    *via DIRECT, Ve 3, [0/0], 9m11s, local, tag 0
```

Configuring static inter-VRF route leaking

Use the following procedure to configure static inter-VRF route leaking.

ATTENTION
Static inter-VRF route leaking is a feature that should be deployed only by an advanced user.

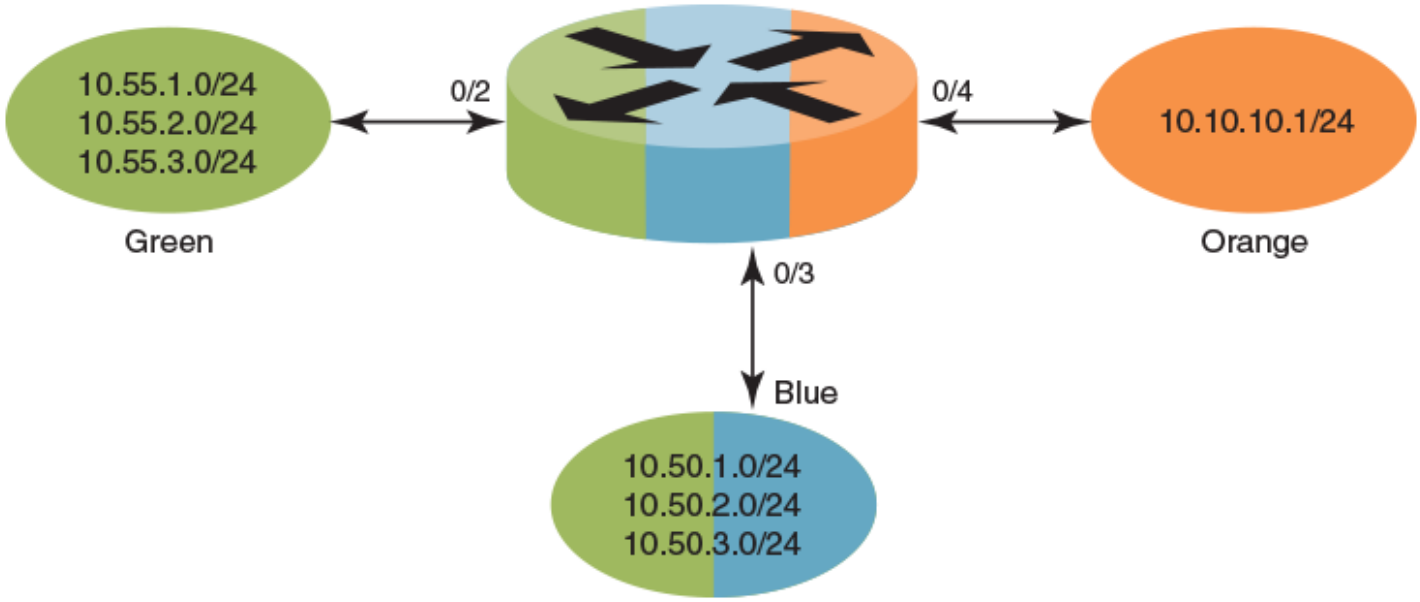
1. Create the VRF instances you want to be the leaker (source VRF) and where the route is being leaked to (destination VRF).
2. Specify the interface for the source VRF and map it to the source VRF.
3. Enter the IP address/mask to be used for this VRF instance.
4. Specify the interface you want to be the destination VRF and map it to the destination VRF.
5. Specify the IP address/mask to receive the leaked route.
6. Change the configuration mode to source VRF address-family context.
7. Configure the route to be leaked, specifying the route prefix, the next-hop VRF name as the destination VRF and the next hop to the destination VRF.
8. Optional: (Optional) For bidirectional inter-VRF route leaking, repeat the above steps, but swap the source and destination addresses.

Example of static inter-VRF route leaking

In this example, one of the static routes in VRF "Blue" (10.50.2.0/24) is being allowed to communicate with one in VRF "Green" (10.55.2.0/24).

The ovals represent virtual partitions (VRFs) in the router. The destination VRF ("Green") is where the route is being leaked to, and the source VRF ("Blue") is where the route is being leaked from.

FIGURE 80 Static inter-VRF route leaking



1. In global configuration mode, create and configure VRF "Green".

```
device(config)# vrf Green
device(conf-vrf-Green)# address-family ipv4 unicast
device(vrf-Green-ipv4-unicast)#
```

2. Repeat the above for VRF "Blue".

```
device(config)# vrf Blue
device(conf-vrf-Blue)# address-family ipv4 unicast
device(vrf-Blue-ipv4-unicast)#
```

3. Configure an interface in the destination VRF "Green" by using the using the **vrf forwarding** command and configuring a corresponding IP address and subnet mask.

```
device(config)# interface eth 0/2
device(conf-eth-0/2)# vrf forwarding Green
device(conf-eth-0/2)# ip address 10.55.1.2/24
```

4. Repeat the above for the source VRF "Blue", with an appropriate interface and network.

```
device(config)# interface eth 0/3
device(conf-eth-0/3)# vrf forwarding Blue
device(conf-eth-0/3)# ip address 10.50.1.2/24
```

5. Enter address-family IPv4 unicast configuration mode for the source VRF address family context for configuring static route leak.

```
device(config)# vrf Blue
device(conf-vrf-Blue)# address-family ipv4 unicast
```

6. Configure route leaking for a network (using the IP address and subnet mask), by specifying the destination next-hop VRF instance and the next hop in the destination VRF.

   NOTE
   The destination VRF can also be a specific port on an Ethernet interface. Refer to the *Extreme SLX-OS Command Reference* for details on the **ip route next-hop-vrf** command.

```
device(vrf-Blue-ipv4-unicast)# ip route 10.55.2.0/24 next-hop-vrf Green 10.55.1.1
```

7. Configure route leaking for the default VRF for a network (using the IP address and subnet mask), by specifying the destination next-hop VRF instance and the default-vrf in the destination VRF.

```
device(vrf-Blue-ipv4-unicast)# ip route 20.0.0.0/24  next-hop-vrf default-vrf 10.1.1.1
```

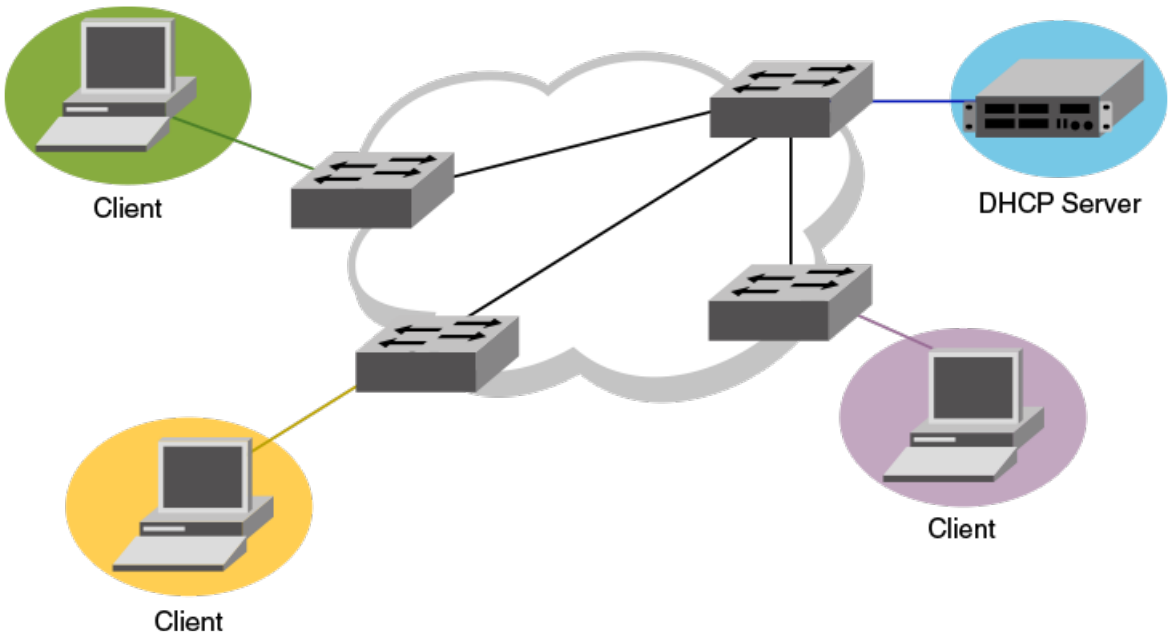
8. (Optional) For bidirectional route-leak traffic, you can also configure route leaking from VRF "Green" to VRF "Blue".

Inter-VRF route leaking and DHCP relay

In a DHCP relay setting, route leaking is controlled through a single DHCP server (which may be on a different VRF); this permits multiple VRFs to communicate with that server, something that would normally not be permitted. DHCP relay deployments in a data center can use Inter-VRF route leaking to achieve server consolidation; this permits clients in multiple VRFs to communicate with a single DHCP server in a different VRF (normally this is not permitted, as VRFs provide route/traffic isolation).

The illustration below shows four VRFs, with three of them connecting to the fourth for DHCP services. (For more information on working with DHCP IP Relay, refer to the "DHCPv4" chapter.)

FIGURE 81 Inter-VRF route leaking example for connecting clients to a DHCP server in a different VRF.



The following example shows setting up Inter-VRF route leaking and DHCP between the red VRF and the blue VRF.

NOTE
Inter-VRF route leaking supports both IPv4 and IPv6. Use the **ip address** and **ip route** commands for IPv4 and the **ipv6 address** and **ipv6 route** commands for IPv6. These commands support IP addresses, Ethernet interfaces, and virtual Ethernet (VE) interfaces for the leak destination. Refer to the *Extreme SLX-OS Command Reference*.

1. Configure VRF forwarding on a VE interface.

```
device(config)# interface ve 100
device(config-if-Ve-100)# no shutdown
device(config-if-Ve-100)# vrf forwarding red
     <- interface is in VRF "red" ->
device(config-if-Ve-100)# ip address 10.1.1.1/24
device(config-if--Ve-100)# ip dhcp relay address 20.1.1.2 use-vrf blue
     <- server is in VRF "blue" ->
```

2. Configure the leaked route on VRF "red".

```
device(config)# vrf red
device(conf-vrf-red)# address-family ipv4 unicast
device(vrf-red-ipv4-unicast)# max-route
device(vrf-red-ipv4-unicast)# ip route 20.1.1.2/32 next-hop-vrf blue 20.2.1.2
```

Configuring dynamic inter-VRF route leaking

Use the following basic procedure to configure dynamic Inter-VRF route leaking.

ATTENTION
Dynamic inter-VRF route leaking is a feature that should be deployed only by an advanced user.

NOTE
Note the following limitations and considerations for route leaking:

- Leaked routes will not be leaked again.
- Control plane protocols cannot run on leaked routes.
- Leaking the same prefix across VRFs is not supported. That is, a given prefix can be present in multiple VRFs, but it should not be leaked from one VRF to another. The behavior in such a case will be inconsistent.

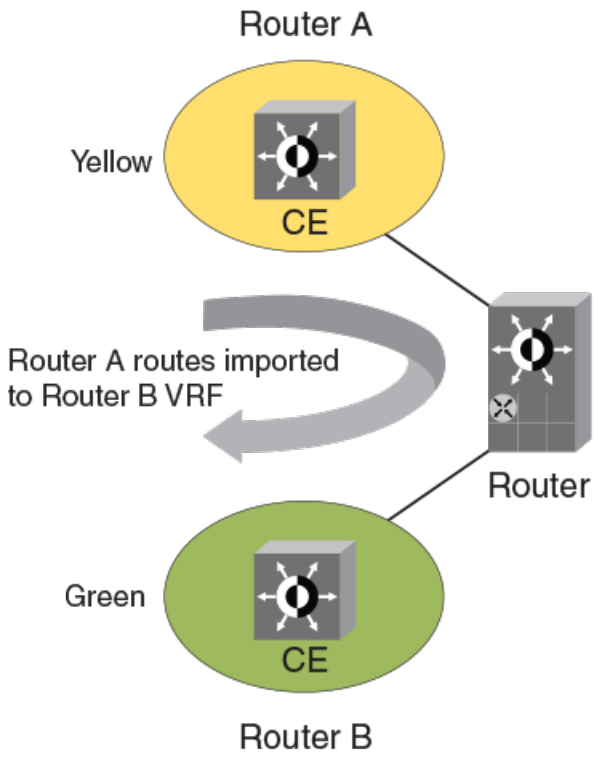
1. Configure the VRF instances you want to be the leaker (source VRF) and where the route is being leaked to (destination VRF).
2. Specify the interface for the source VRF and map it to the source VRF.
3. Enter the IP address/mask to use for this VRF instance.
4. Specify the interface you want to be the destination VRF and map it to the destination VRF.
5. Specify the IP address/mask to receive the leak.
6. Configure the route map and associated prefix-list.
7. Change the configuration mode to destination VRF address family context. (IPv4 and IPv6 are supported.)
8. Configure the import command, specifying the source VRF and route map to be leaked.
9. (Optional) You can leak BGP or OSPF routes that were learned by the source VRF into the destination VRF. Static routes can also be leaked.

Example of Dynamic Inter-VRF route leaking

In this example, a route map called "import-map" has match criteria specified as prefixes that can be learned by means of routing protocols such as OSPF or BGP.

The figure below depicts a typical dynamic route-leaking scenario. VRFs “Yellow” and “Green” are virtual partitions in the same router. The destination VRF ("Green") is where the route is being leaked to, and the source VRF ("Yellow") is where the route is being leaked from. In this example, IPv4 is used.

FIGURE 82 Dynamic inter-VRF route leaking



1. Configure VRF "Green".

```
device(config)# vrf Green
device(config-vrf-Green)# address-family ipv4 unicast
```

2. Configure VRF "Yellow".

```
device(config)# vrf Yellow
device(config-vrf-Yellow)# address-family ipv4 unicast
```

3. Configure an IPv4 prefix list, named "import-prefix" in this example.

```
device(config)# ip prefix-list import-prefix permit 10.2.3.0/24
device(config)# ip prefix-list import-prefix permit 10.1.2.0/24
```

4. Configure a route map with "match" conditions, including metric and tag for matched routes, and then import it.

```
device(config)# route-map import-map permit 10
device(config-route-map-import-map/permit/10)# match metric 10
device(config-route-map-import-map/permit/10)# match tag 10
device(config-route-map-import-map/permit/10)# match ip address prefix-list import-prefix
```

5. Import the desired route map for the specified VRF.

```
device(config)# vrf Green
device(config-vrf-Green)# address-family ipv4 unicast
device(vrf-Green-ipv4-unicast)# ip import routes Yellow route-map import-map
```

6. (Optional) Redistribute any routes learned by OSPF (or BGP) in the source VRF into the destination VRF. The following shows an OSPF example.

```
device(config-ipv4-unicast)# exit
device(config-vrf-Green)# exit
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# redistribute ospf
```

Commands for dynamic inter-VRF route leaking

Commands you can use to import dynamic routes for Inter-VRF leaking are included in the following table and described in detail in the *Extreme SLX-OS Command Reference*.

TABLE 46 Dynamic inter-VRF route leaking commands

Command	Description	Mode
ip import routes *VRF_name* **route-map** *rmap_name*	Leaks IPv4 routes from a specified VRF to the default VRF, based on match criteria defined in the specified route-map.	Global configuration
ip import routes *VRF_name* **route-map** *rmap_name*	Leaks IPv4 routes from one VRF to another VRF, based on match criteria defined in the specified route-map.	VRF address family configuration
ipv6 import routes *VRF_name* **route-map** *rmap_name*	Leaks IPv6 routes from a specified VRF to the default VRF, based on match criteria defined in the specified route-map.	Global configuration
ipv6 import routes *VRF_name* **route-map** *rmap_name*	Leaks IPv6 routes from one VRF to another VRF, based on match criteria defined in the specified route-map.	VRF address family configuration
show ip route import	Displays the IPv4 routes imported to a specified VRF.	Privileged EXEC
show ipv6 route import	Displays the IPv6 routes imported to a specified VRF.	Privileged EXEC
redistribute ospf	Redistributes leaked OSPFv3 (or OSPF v2) routes that were imported to another VRF into OSPF of this VRF instance.	OSPF VRF router configuration
redistribute bgp	Redistributes leaked BGP routes that were imported to another VRF into BGP of this VRF instance.	• Address-family ipv4 unicast • Address-family ipv6 unicast • Address-family ipv4 unicast vrf • Address-family ipv6 unicast vrf

NOTE

The **redistribute** commands enable OSPF or BGP to take the routes leaked from other VRFs and advertise them to peers. This enables the propagation of reachability information to other routers in the network for traffic forwarding.

Internet route scaling

Internet route scaling (OptiScale™ for Internet Routing) helps to ensure reachability for all networks in the internet routing table and to eliminate unnecessary table entries.

OptiScale for Internet Routing creates an enhanced route scale profile using the following methods:

- FIB (Forwarding Information Base) optimization: Protocol routes are downloaded to the RIB (Routing Information Base), which then downloads them to the FIB. Initial route optimization eliminates redundant next-hop entries.
- Hardware table optimization: Routes with common prefixes are stored in separate tables. The Longest Exact Match (LEM) table handles common prefixes. The Longest Prefix Match (LPM) table retains enough headroom to accommodate route entries that are added.

OptiScale for Internet Routing is supported as follows:

- It cannot coexist with uRPF (unicast Reverse Path Forwarding).
- (SLX 9850, SLX 9640) It is supported in the default and non-default VRF. The SLX 9850 supports the default VRF only when OptiScale is enabled.
- (SLX 9540) It is supported only in the default VRF. The SLX 9540 supports the default VRF only when OptiScale is enabled.

For more information, see the *OptiScale™ for Internet Routing* white paper at https://www.extremenetworks.com/resources/white-paper/optiscale-for-internet-routing-internet-scale-routing-on-slx-border-leaf-and-border-platforms/.

Configuring profile route

Follow these steps to specify the route mode.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **hardware** command to access configuration hardware mode.

```
device(config)# hardware
```

3. Enter the **profile** command to view the possible completion.

```
device(config-hardward)# profile?

Possible completions:
  counters      Select counter profile
  lag           Select LAG profile
  route         Select route mode or profile type
  tcam          Select TCAM profile type
```

4. Enter the **profile route** command to view the possible completion.

```
device(config-hardware)# profile route ?
Possible completions:
  [default]
  default       Default routing mode (??? Routes)
  enhanced      Enhanced routing mode (supporting 1M routes)
```

5. Enter the **profile route enchaned** command to view the possible completion.

```
device(config-hardware)# profile route enhanced ?
Possible completions:
  ipv4-fib-compressed    Enable/disable ipv4-fib-compression
  ipv6-fib-compressed    Enable/disable ipv6-fib-compression
  hw-optimized           Enable/disable HW optimization
```

6. Enter the **profile route enhanced ipv4-fib-compressed** command to view the possible completion.

```
device(config-hardware)# profile route enhanced ipv4-fib-compressed ?
Possible completions:
  [off]
  Off                    Disable ipv4-fib-compressed
  On                     Enable ipv4-fib-compressed
```

7. Enter the **profile route enhanced ipv6-fib-compressed** command to view the possible completion.

```
device(config-hardware)# profile route enhanced ipv6-fib-compressed ?
Possible completions:
  [off]
  Off                    Disable ipv6-fib-compressed
  On                     Enable ipv6-fib-compressed
```

8. Enter the **profile route enhanced hw-optimized** command to view the possible completion.

```
device(config-hardware)# profile route enhanced hw-optimized ?
Possible completions:
  [off]
  Off                    Disable HW optimization
  On                     Enable HW optimization
```

VRRPv2

- VRRPv2 overview ... 623
- Enabling a master VRRP device ... 632
- Enabling an owner VRRP device ... 636
- Enabling a backup VRRP device ... 638
- Enabling a backup VRRP device ... 639
- Configuring simple text authentication on VRRP interfaces ... 640
- Configuring MD5 authentication on VRRP interfaces ... 641
- Abdicating VRRP master device status ... 642
- VRRP multigroup clusters ... 643
- Tracked ports and track priority with VRRP and VRRP-E ... 650
- VRRP owner preemption and interface tracking ... 655
- VRRP backup preemption ... 656
- Accept mode for backup VRRP devices ... 658
- Accept mode for backup VRRP devices ... 659
- Virtual router MAC address ... 660
- Suppressing RIP route advertisements on VRRP backup devices ... 663
- VRRP-Ev2 overview ... 664
- Enabling a VRRP-E device ... 664
- Enabling a VRRP-E device ... 666
- Configuring MD5 authentication on IPv4 VRRP-E interfaces ... 667
- Track routes and track priority with VRRP-E ... 669
- VRRP-E load-balancing using short-path forwarding ... 672
- VRRP-E slow start timer ... 676
- Multiple virtual IP address support for VRRP-E ... 677
- VRRP-E scaling using logical groups ... 680
- Displaying VRRPv2 information ... 682
- Clearing VRRPv2 statistics ... 687

VRRPv2 overview

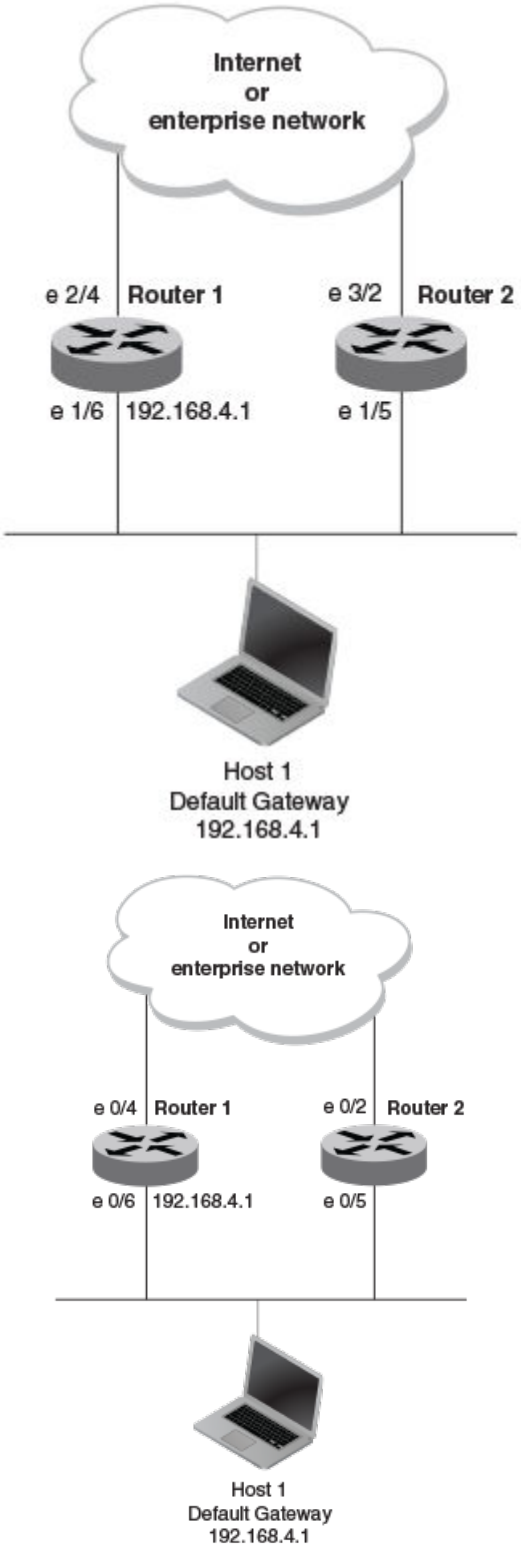
Virtual Router Redundancy Protocol (VRRP) is an election protocol that provides redundancy to routers within a Local Area Network (LAN).

VRRP was designed to eliminate a single point of failure in a static default-route environment by dynamically assigning virtual IP routers to participating hosts. A virtual router is a collection of physical routers whose interfaces must belong to the same IP subnet. A virtual router ID (VRID) is assigned to each virtual router, but there is no restriction against reusing a VRID with a different address mapping on different LANs.

> **NOTE**
> VRRP extended (VRRP-E) is an extended version of the VRRP protocol. Extreme Networks developed VRRP-E as a proprietary protocol to address some limitations in standards-based VRRP.

Before examining more details about how VRRP works, it is useful to see why VRRP was developed to solve the issue of a single point of failure.

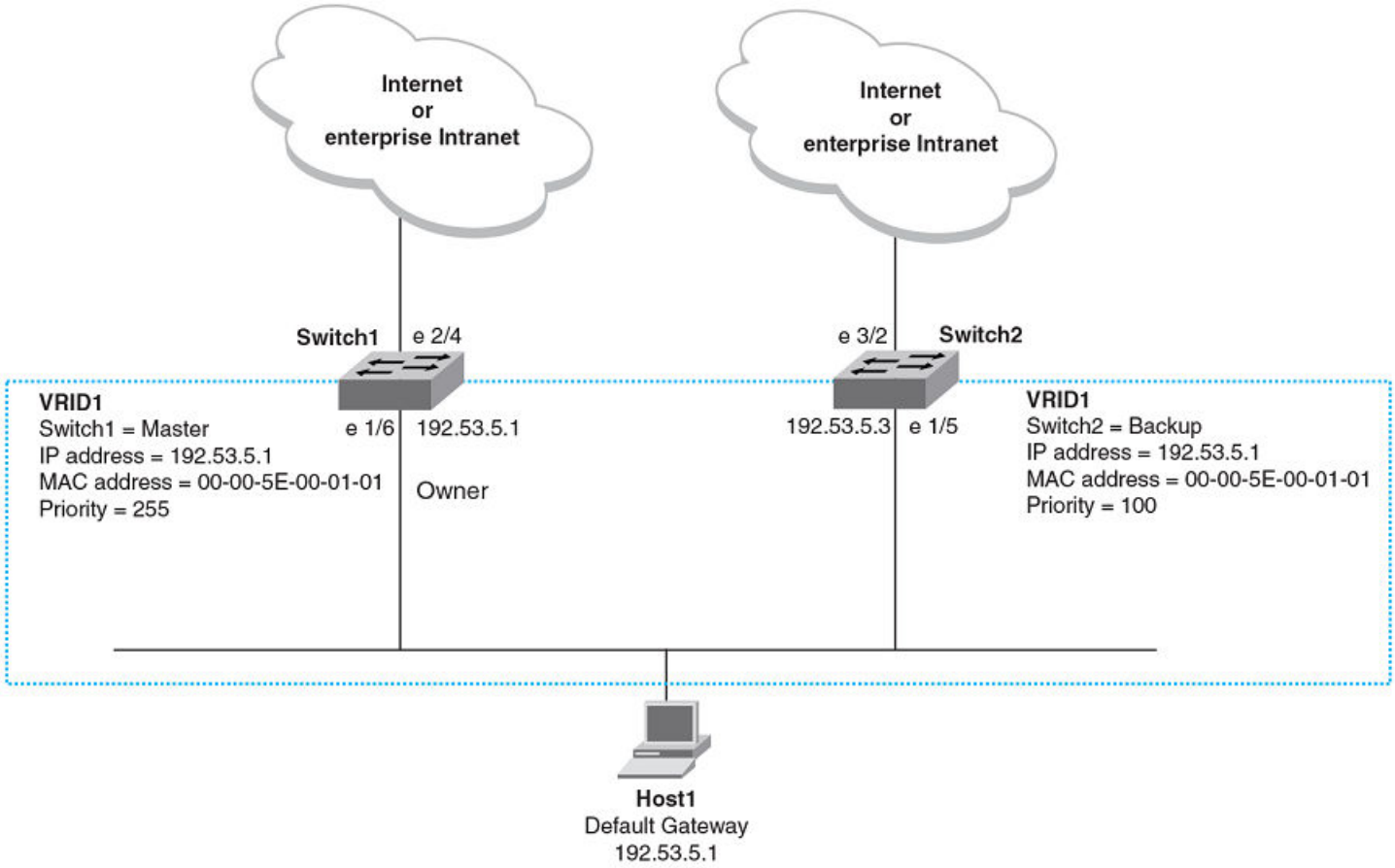
FIGURE 83 Single point of failure with Device 1 being the Host1 default gateway

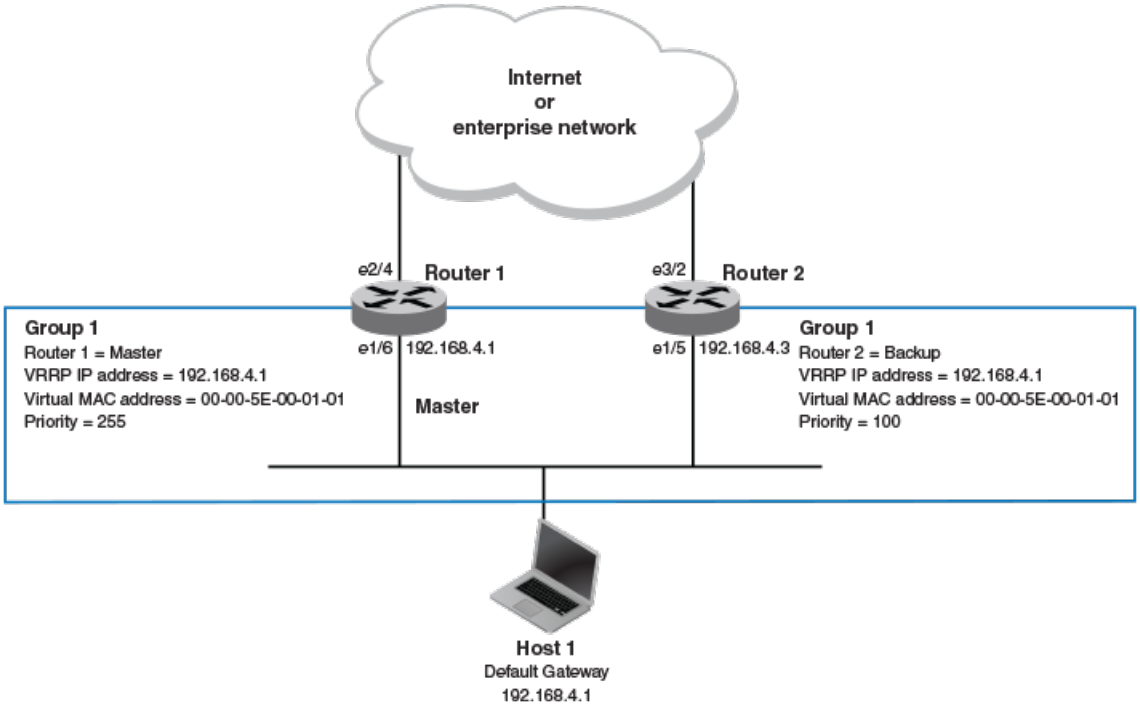


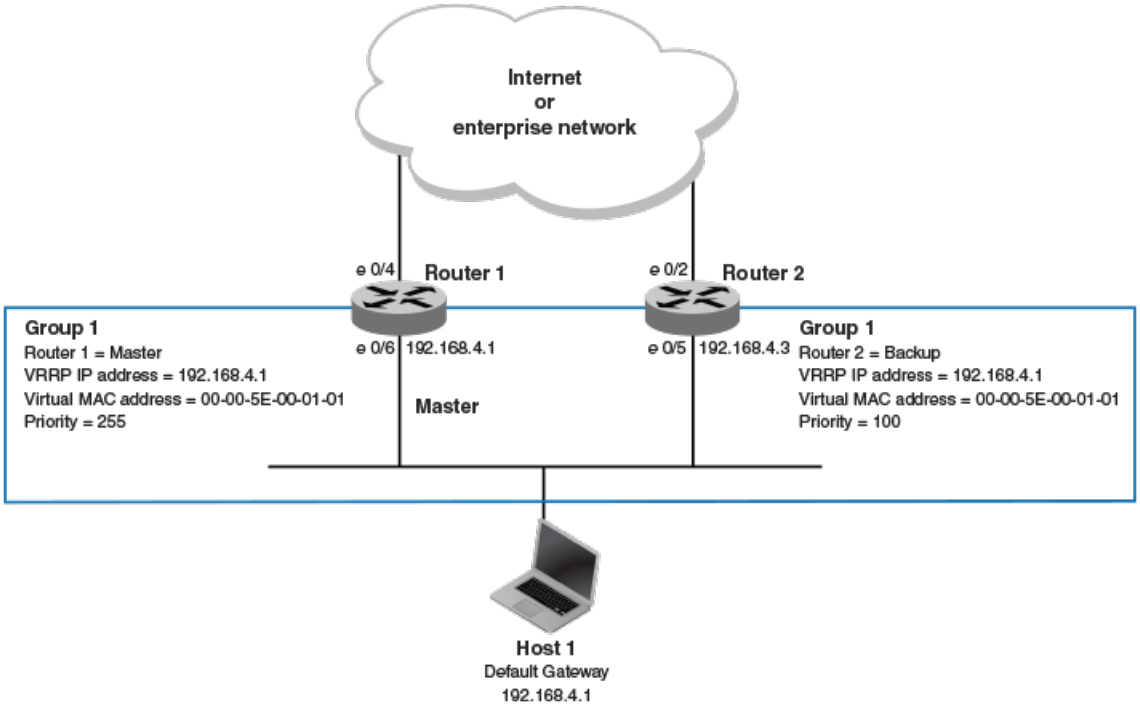
To connect to the Internet or an internal intranet Host 1, in the figure, uses the IP address of 192.168.4.1 on Router 1 as its default gateway. If this interface goes down, Host 1 is cut off from the rest of the network. Router 1 is a single point of failure for Host 1 to access other networks. In small networks, the administrative burden of configuring Router 2 as the new default gateway is not an issue, but in larger networks reconfiguring default gateways is impractical. Configuring a VRRP virtual router on Router 1 and Router 2 provides a redundant path for the hosts. VRRP allows you to provide alternate router paths for a host without changing the IP address or MAC address by which the host knows its gateway.

To illustrate how VRRP works, the following figure shows the same network, but a VRRP virtual router is configured on the two physical routers, Router 1 and Router 2. This virtual router provides redundant network access for Host 1. If Router 1 were to fail, Router 2 would provide the default gateway out of the subnet.

FIGURE 84 Devices configured as VRRP virtual routers for redundant network access for Host 1







The blue rectangle in the figure represents a VRRP virtual router. When you configure a virtual router, one of the configuration parameters is a group number (also known as a virtual router ID or VRID), which can be a number from 1 through 255. The virtual router is identified with a group, and within the VRRP group, there is one physical device that forwards packets for the virtual router and this is called a master VRRP device. The VRRP master device may be a Layer 3 switch or a router.

In VRRP, one of the physical IP addresses is configured as the IP address of the virtual router, the virtual IP address. The device on which the virtual IP address is assigned becomes the VRRP owner, and this device responds to packets addressed to any of the IP addresses in the virtual router group. The owner device becomes the master VRRP device by default and is assigned the highest priority. Backup devices are configured as members of the virtual router group, and, if the master device goes offline, one of the backup devices assumes the role of the master device.

NOTE
VRRP operation is independent of BGP4 and OSPF. Their operation is unaffected when VRRP is enabled on the same interface as BGP4 or OSPF.

NOTE
VRRP operation is independent of BGP4, OSPF, and ISIS. Their operation is unaffected when VRRP is enabled on the same interface as BGP4, OSPF, or ISIS.

NOTE
VRRP operation is independent of BGP4 and OSPF. Their operation is unaffected when VRRP is enabled on the same interface as BGP4 or OSPF.

NOTE
VRRP operation is independent of BGP4, OSPF, and RIP. Their operation is unaffected when VRRP is enabled on the same interface as BGP4, OSPF, and RIP.

VRRP terminology

Before implementing VRRP in your network, you must understand some key terms and definitions.

The following VRRP-related terms are in logical order, not alphabetic order:

Virtual router	A collection of physical routers that can use VRRP to provide redundancy to routers within a LAN.
Virtual router ID	A group of physical routers that are assigned to the same virtual router ID (VRID).
Virtual router address	The virtual router IP address must belong to the same subnet as a real IP address configured on the VRRP interface, and it can be the same as a real IP address configured on the VRRP interface. The virtual router whose virtual IP address is the same as a real IP address is the IP address owner and the default master.
Owner	The owner is the physical router whose real interface IP address is the IP address that you assign to the virtual router. The owner responds to packets addressed to any of the IP addresses in the corresponding virtual router. The owner, by default, is the master and has the highest priority (255).
Master	The physical router that responds to packets addressed to any of the IP addresses in the corresponding virtual router. For VRRP, if the physical router whose real interface IP address is the IP address of the virtual router, then this physical router is always the master.
Backup	Routers that belong to a virtual router, but are not the master. If the master becomes unavailable, the backup router with the highest priority (a configurable value) becomes the new master. By default, routers are given a priority of 100.

Virtual router	A collection of physical routers that can use VRRP to provide redundancy to routers within a LAN.
Virtual router group	A group of physical routers that are assigned to the same virtual router.
Virtual router address	The virtual router IP address must belong to the same subnet as a real IP address configured on the VRRP interface, and it can be the same as a real IP address configured on the VRRP interface. The virtual router whose virtual IP address is the same as a real IP address is the IP address owner and the default master.
Owner	The owner is the physical router whose real interface IP address is the IP address that you assign to the virtual router. The owner responds to packets addressed to any of the IP addresses in the corresponding virtual router. The owner, by default, is the master and has the highest priority (255).

Master The physical router that responds to packets addressed to any of the IP addresses in the corresponding virtual router. For VRRP, if the physical router whose real interface IP address is the IP address of the virtual router, then this physical router is always the master.

Backup Routers that belong to a virtual router, but are not the master. If the master becomes unavailable, the backup router with the highest priority (a configurable value) becomes the new master. By default, routers are given a priority of 100.

VRRPv2 limitations on VDX devices

The implementation of VRRPv2 varies across the VDX products.

Virtual routers must be configured in a Virtual Cluster Switching (VCS) environment.

Only IPv4 support is provided in VRRPv2. VRRPv3 supports both IPv4 and IPv6.

Supported ports:

- For VRRP—fortygigabitethernet, tengigabitethernet, gigabitethernet, port-channel, and ve
- For VRRP-E—ve ports only

System Resource	VDX 87xx	VDX 6740, 6740T, 6740T-1GT	VDX 6940-36Q, 6940-144S
Max # of VRRP and VRRP-E sessions	1024	255	512

The maximum number of virtual IP addresses per virtual router session is 32 for VRRP, and one for VRRP-E.

SLX-OS VRRP virtual router MAC address limitations

Virtual MAC (VMAC) considerations on the SLX hardware affect the number of VRRP virtual router IDs (VRIDs) available for the VRRP and VRRP Extended (VRRP-E) sessions.

When you configure a virtual routing ID (VRID), the software automatically uses the MAC address as the MAC address of the virtual router. The first five octets of the address are the standard MAC prefix for VRRP packets. The last octet is the VRID.

When the virtual router becomes the master router, it broadcasts a gratuitous ARP (GARP) request containing the virtual router's MAC address for each IP address associated with the virtual router. Hosts use the MAC address of the virtual router in routed traffic they send to their default IP gateway.

You can manually configure a unique virtual MAC address for each IPv4 and IPv6 VRRP instance per VRID. If there is no manually configured virtual MAC address for a VRRP instance, the system automatically assigns one.

TABLE 47 VRRP/VRRP-E VRID to VMAC mapping

Protocol	VMAC	#VMACs supported	# VRIDs supported
Standard IPv4 VRRP (v2/v3)	00-00-5E-00-01-{VRID}	16	16
Standard IPv6 VRRP (v3)	00-00-5E-00-02-{VRID}	16	16
SLX-OS IPv4 VRRP-E (v2/v3)	02:E0:52:00:01:<1-byte-mvrid>	16 unique VMACs	mvrid is (((vrid-1) ÷16) + 1) with VRID ranging between 1 to 255
SLX-OS IPv6 VRRP-E (v3)	02:E0:52:00:02:<1-byte-mvrid>	16 unique VMACs	mvrid is (((vrid-1) ÷ 16) + 1) with VRID ranging between 1 to 255

NOTE

The 16 unique VMACs per hardware chip have to be used across all interfaces, virtual routing forwarding instances (VRFs), VRRP and VRRP-E sessions. VRIDs must not be used between VRRP/VRRP-E IPv4 and IPv6 sessions.

SLX-OS VRRP and VRRP-E interoperability

While VRRP follows RFC standards and can be used across devices from multiple vendors, Extreme Networks has its proprietary extension of VRRP, VRRP Extended (VRRP-E) which has strict interoperability rules across its devices.

In the standard SLX-OS VRRP implementation, the first five octets of the virtual MAC (VMAC) address are the standard MAC prefix for VRRP packets. The last octet is the virtual router ID (VRID). The only requirement for interoperability across other Extreme platform devices or other vendor devices is to ensure that the VRID is in the range of 1 to 16.

SLX-OS VRRP interoperability with Extreme or non-Extreme devices

The virtual router ID (VRID) must be in the range of 1 to 16.

SLX-OS VRRP-E interoperability with Extreme VDX devices

- The virtual router ID (VRID) must be in the range of 1 to 16.
- You must manually configure the VMAC on the Extreme VDX device for shared VRRP-E session with an SLX device.
- You must manually configure the VMAC on the SLX device for shared VRRP-E session with a VDX device.

SLX-OS VRRP-E interoperability with Extreme MLXe devices

- The virtual router ID (VRID) must be in the range of 1 to 16.
- You must manually configure the VMAC on the Extreme MLXe device using the MAC address from the SLX device VMAC for shared VRRP-E session with an SLX device.
- You must manually configure the VMAC on the SLX device for shared VRRP-E session with an Extreme MLXe device.

VRRP hold timer

The hold timer delays the preemption of a master VRRP device by a high-priority backup device.

A hold timer is used when a VRRP-enabled device that was previously a master device failed, but is now back up. This restored device now has a higher priority than the current VRRP master device, and VRRP normally triggers an immediate switchover. In this situation, it is possible that not all software components on the backup device have converged yet. The hold timer can enforce a waiting period before the higher-priority backup device assumes the role of master VRRP device again. The timer must be set to a number greater than 0 seconds for this functionality to take effect.

Hold timer functionality is supported in both version 2 and version 3 of VRRP and VRRP-E.

VRRP interval timers

Various timers for the intervals between hello messages sent between devices running VRRP can be configured.

Hello intervals

Hello messages are sent from the master VRRP device to the backup devices. The purpose of the hello messages is to determine that the master device is still online. If the backup devices stop receiving hello messages for a period of time, as defined by the dead (or master-down-interval) interval, the backup devices assume that the master device is offline. When the master device is offline, the backup device with the highest priority assumes the role of the master device.

NOTE
The hello intervals must be set to the same value on both owner and backup devices for the same VRID.

Dead interval

The dead interval is defined as the period of time for which backup devices wait for a hello message from the master device before assuming that the master device is offline. An immediate switchover to the backup device with the highest priority is triggered after the dead interval expires and there is no hello message from the master device. If a value for the dead interval is not configured, the default value is calculated as three times the hello interval plus the skew time. Skew time is defined as (256 – priority)/256.

NOTE
The dead interval must be set to the same value on both owner and backup devices for the same VRID.

Backup hello message state and interval

By default, backup devices do not send hello messages to advertise themselves to the master device. Hello messages from backup devices can be activated, and the messages are sent at 60-second intervals, by default. The interval between the backup hello messages can be modified.

VRRP authentication

The VRRP authentication type is not a parameter specific to the virtual router. VRRP uses the authentication type associated with the interfaces on which the virtual router is defined.

If your interfaces do not use authentication, neither does VRRP. For example, if you configure your device interfaces to use an MD5 password to authenticate traffic, VRRP uses the same MD5 password, and VRRP packets that do not contain the password are dropped.

In summary, if the interfaces on which you configure the virtual router use authentication, the VRRP or VRRP Extended (VRRP-E) packets on those interfaces must use the same authentication. The following VRRP and VRRP-E authentication types are supported:

- No authentication—The interfaces do not use authentication. This authentication type is the default for VRRP and VRRP-E.
- Simple—The interfaces use a simple text string as a password in packets that they send. If the interfaces use simple password authentication, the virtual router configured on the interfaces must use the same authentication type and the same password.
- MD5—This method of authentication ensures that the packet is authentic and cannot be modified in transit. Syslog and SNMP traps are generated when a packet is dropped due to MD5 authentication failure. MD5 authentication is supported only in VRRP-E, and the device configuration is unique on a per-interface basis. The MD5 authentication configuration on an interface takes effect for all VRRP-E virtual routers configured on a particular interface.

NOTE
Authentication is not supported for VRRPv3.

VRRP master device abdication to backup device

To allow temporary control of a VRRP virtual router ID (VRID) to pass to a backup device, you can force the master device to abdicate to a backup device by setting a lower priority.

Changing the priority of a VRRP master device allows a temporary abdication of the master device status to allow a backup device with a higher priority to assume the master device role. By default, a VRRP owner device has a priority of 255, and the lower priority must be set to a lower priority than at least one of the backup devices associated with the VRID.

When you change the priority of a VRRP owner, the change takes effect only for the current power cycle. The change is not saved to the startup configuration file when you save the configuration, and it is not retained across a reload or reboot. Following a reload or reboot, the VRRP owner again has priority 255.

NOTE
This feature supports IPv4 VRRP only. IPv6 VRRP, VRRP-E, and IPv6 VRRP-E are not supported.

ARP and VRRP control packets

Control packets for ARP and VRRP are handled differently by VRRP and VRRP-E.

Source MAC addresses in VRRP control packets

- VRRP—The virtual MAC address is the source.
- VRRP-E—The physical MAC address is the source.

VRRP control packets

- VRRP—Control packets are IP type 112 (reserved for VRRP), and they are sent to the VRRP multicast address 224.0.0.18.
- VRRP-E—Control packets are UDP packets destined to port 8888, and they are sent to the all-router multicast address 224.0.0.2.

Gratuitous ARP

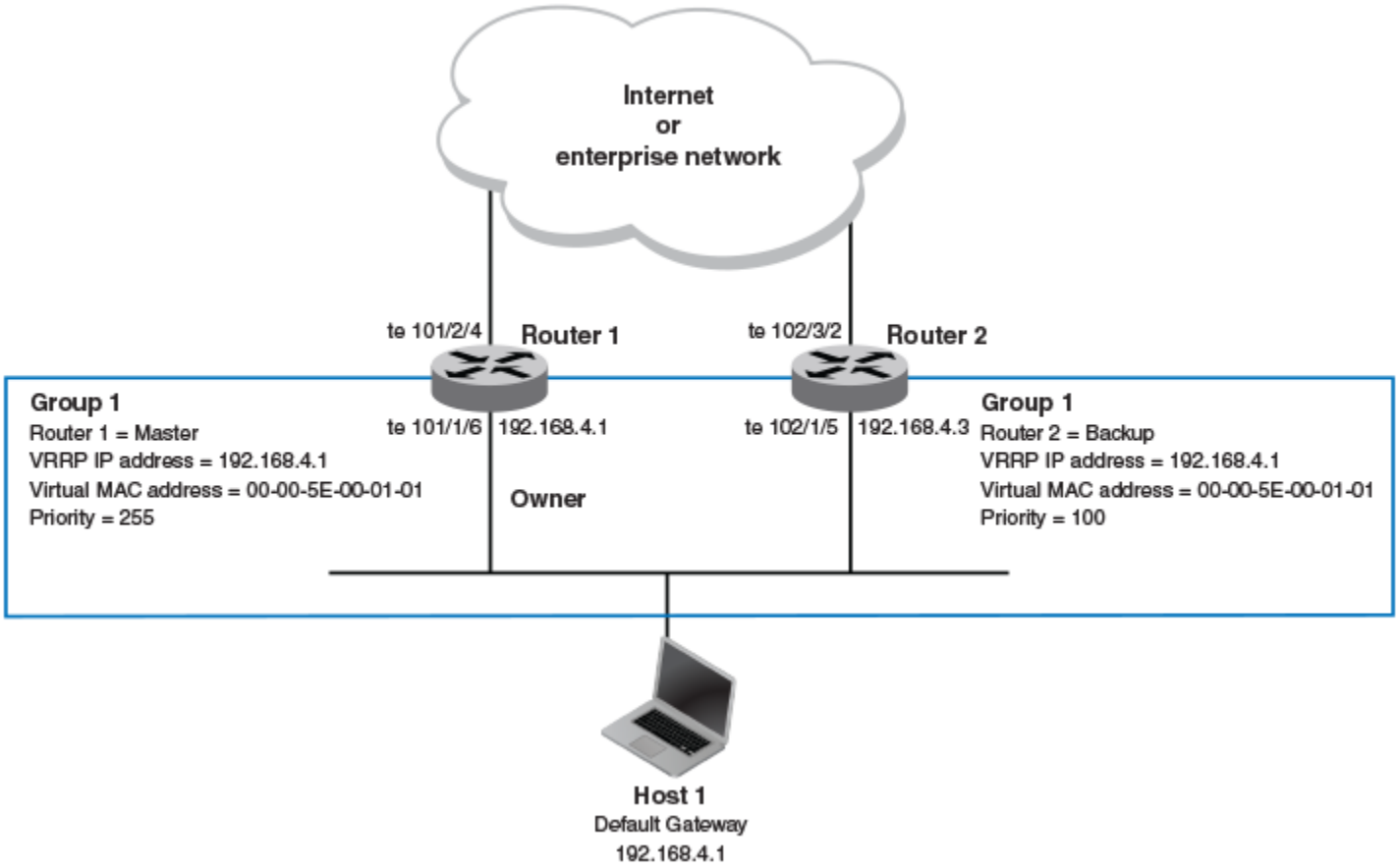
When a VRRP device (either master or backup) sends an ARP request or a reply packet, the MAC address of the sender is the MAC address of the router interface. One exception is if the owner sends an ARP request or a reply packet, in which case the MAC address of the sender is the virtual MAC address. Only the master answers an ARP request for the virtual router IP address. Any backup router that receives this request forwards the request to the master.

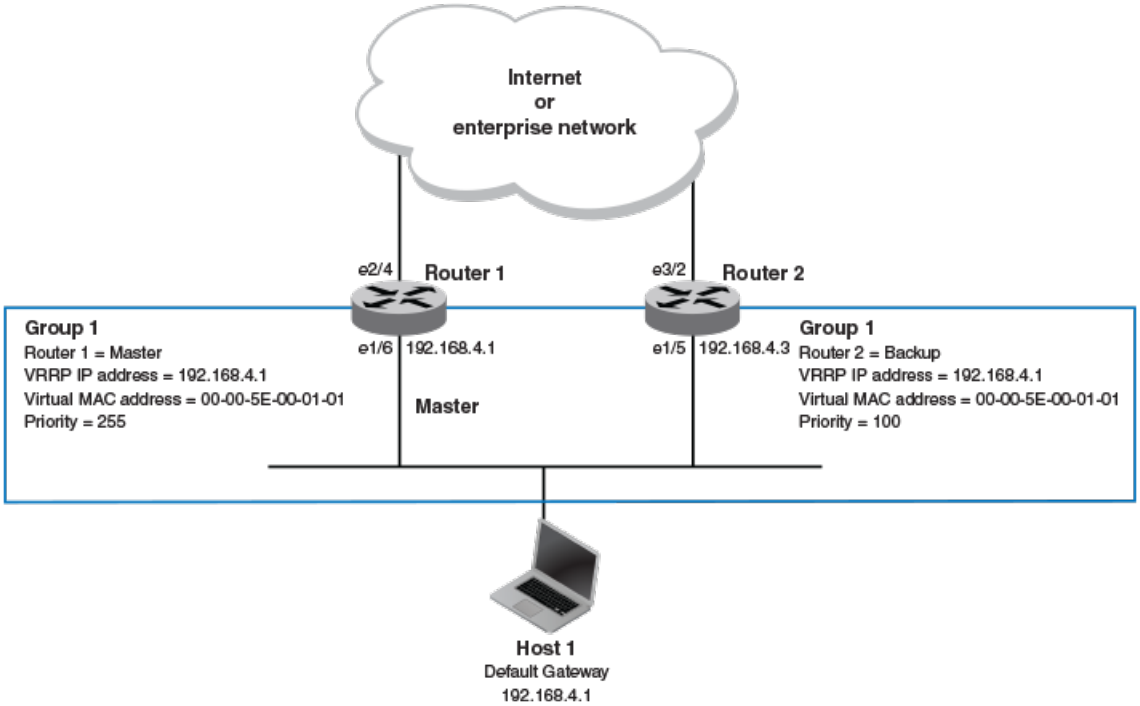
- VRRP—A control message is sent only once when the VRRP device assumes the role of the master.
- VRRP-E—A control message is sent every 2 seconds by the VRRP-E master device because VRRP-E control packets do not use the virtual MAC address.
- VRRP-E—A control message is sent every 30 seconds by the VRRP-E master device because VRRP-E control packets do not use the virtual MAC address.

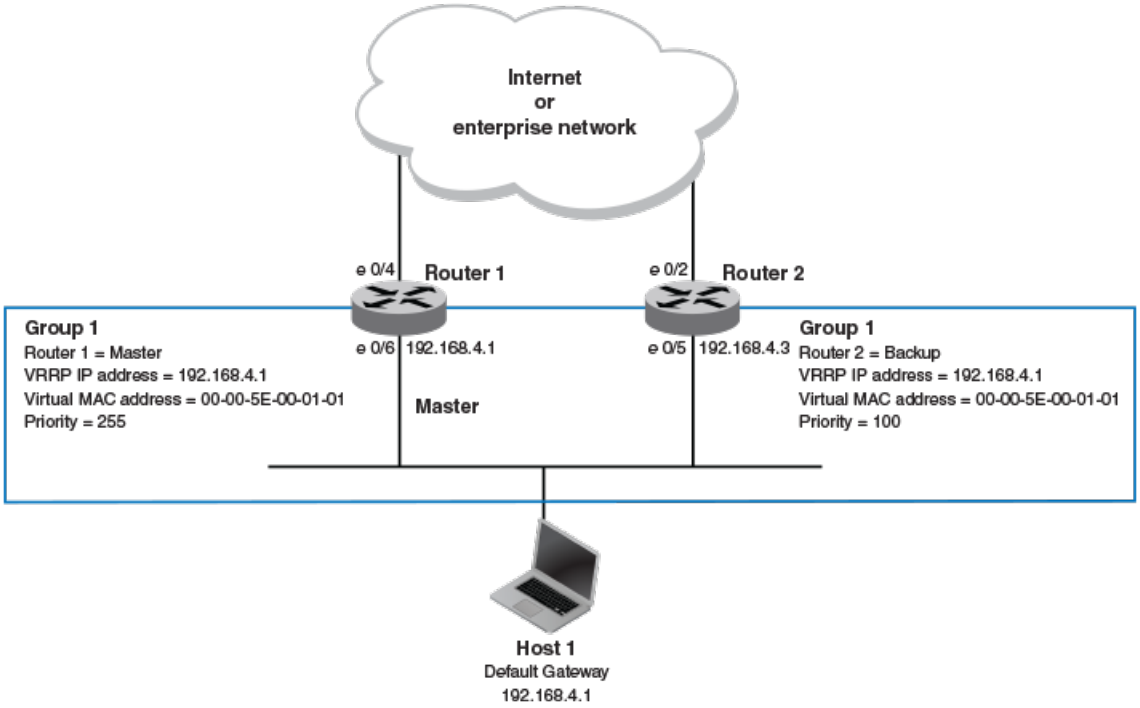
Enabling a master VRRP device

This task is performed on the device that is designated as the master VRRP device. For example, Router 1 is the master VRRP device In the figure that follows.

FIGURE 85 Basic VRRP topology







1. On the device designated as the master VRRP device, and from privileged EXEC mode, enter configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command, using the RBridge ID (which has an asterisk next to it when you run a **do show vcs** command).

```
device(config)# rbridge-id 101
```

3. Globally enable VRRP.

```
device(config-rbridge-id-101)# protocol vrrp
device(config)# protocol vrrp
```

4. Configure the Ethernet interface link for Router 1.

```
device(config-rbridge-id-101)# interface tengigabitethernet 101/1/6
device(config)# interface ethernet 1/6
device(config)# interface ethernet 0/6
```

5. Configure the IP address of the interface.

```
device(config-if-te-101/1/6)# ip address 192.168.4.1/24
device(conf-if-eth-1/6)# ip address 192.168.4.1/24
device(conf-if-eth-0/6)# ip address 192.168.4.1/24
```

6. Assign Router 1 to a group called Group 1.

```
device(config-if-te-101/1/6)# vrrp-group 1
device(conf-if-eth-1/6)# vrrp-group 1
device(conf-if-eth-0/6)# vrrp-group 1
```

NOTE
You can assign a group number in the range of 1 through 255.

7. Assign a virtual router IP address.

```
device(config-vrrp-group-1)# virtual-ip 192.168.4.1
```

NOTE
For VRRP, the physical router whose IP address is the same as the virtual router group IP address becomes the owner and master.

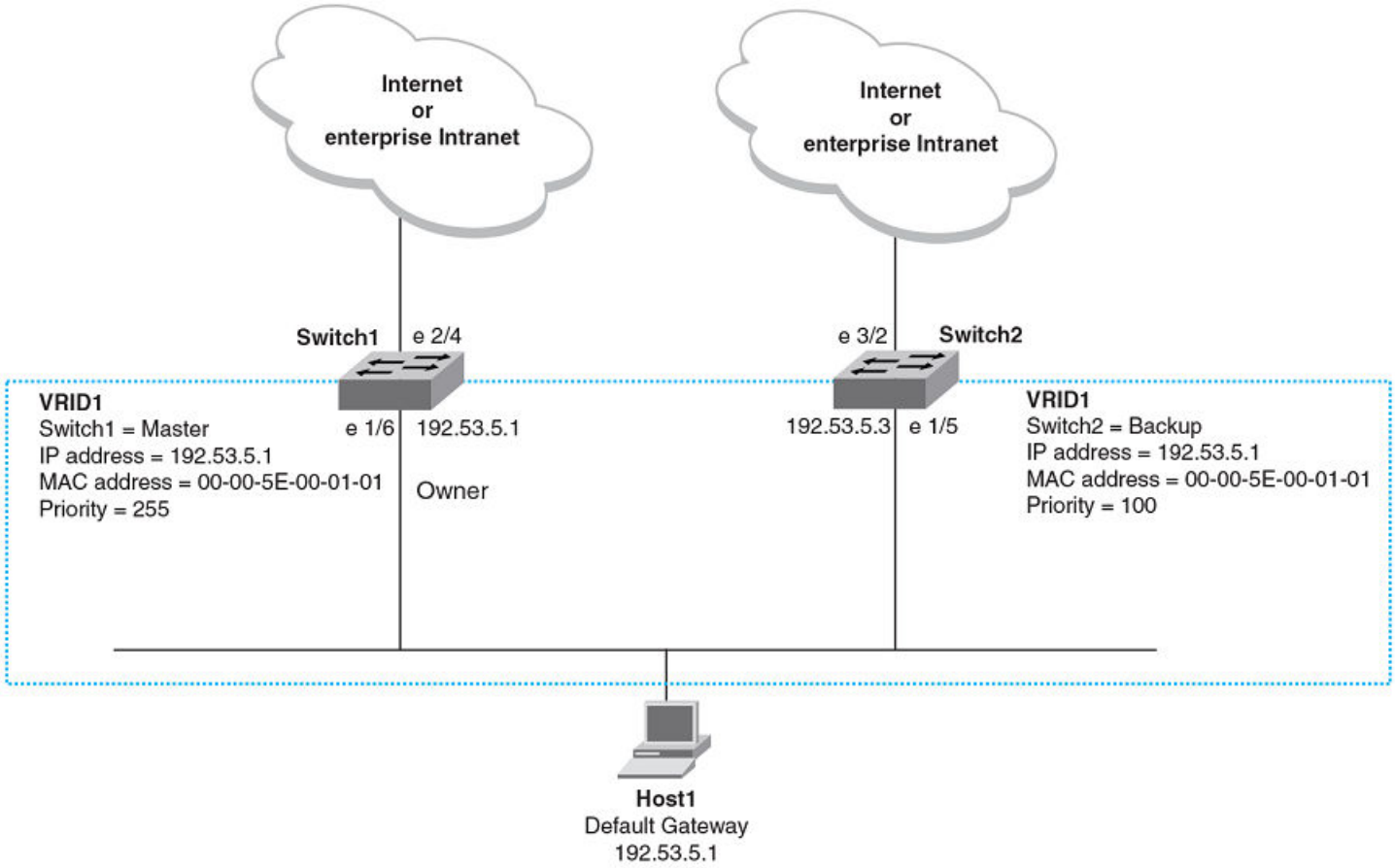
The following example configures a VRRP master device.

```
device# configure
device(config)# rbridge-id 101
device(config-rbridge-id-101)# protocol vrrp
device(config-rbridge-id-101)# interface tengigabitethernet 101/1/6
device(config-if-te-101/1/6)# ip address 192.168.4.1/24
device(config-if-te-101/1/6)# vrrp-group 1
device(config-vrrp-group-1)# virtual-ip 192.168.4.1

device# configure
device(config)# protocol vrrp
device(config)# interface ethernet 1/6
device(conf-if-eth-1/6)# ip address 192.168.4.1/24
device(conf-if-eth-1/6)# vrrp-group 1
device(config-vrrp-group-1)# virtual-ip 192.168.4.1

device# configure
device(config)# protocol vrrp
device(config)# interface ethernet 0/6
device(conf-if-eth-0/6)# ip address 192.168.4.1/24
device(conf-if-eth-0/6)# vrrp-group 1
device(config-vrrp-group-1)# virtual-ip 192.168.4.1
```

Enabling an owner VRRP device

This task is performed on the device that is designated as the owner VRRP device because the IP address of one of its physical interfaces is assigned as the IP address of the virtual router. For example, Router 1 is the owner VRRP device in the figure that follows. For each VRRP session, there are master and backup routers, and the owner router is elected, by default, as the master router.

FIGURE 86 Basic VRRP topology



1. On the device designated as the owner VRRP device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP.

```
device(config)# router vrrp
```

3. Configure the Ethernet interface link for Router 1.

```
device(config)# interface ethernet 1/6
```

4. Configure the IP address of the interface.

```
device(config-if-e1000-1/6)# ip address 10.53.5.1/24
```

5. Assign Router 1 to the virtual router ID (VRID) 1.

```
device(config-if-e1000-1/6)# ip vrrp vrid 1
```

NOTE
You can assign a VRID number in the range of 1 through 255.

6. Designate this router as the VRRP owner device.

```
device(config-if-e1000-1/6-vrid-1)# owner
```

7. Configure the IP address of the VRID.

```
device(config-if-e1000-1/6-vrid-1)# ip-address 10.53.5.1
```

8. Enable the VRRP session.

```
device(config-if-e1000-1/6-vrid-1)# activate
```

The following example configures a VRRP owner device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/6
device(config-if-e1000-1/6)# ip address 10.53.5.1/24
device(config-if-e1000-1/6)# ip vrrp vrid 1
device(config-if-e1000-1/6-vrid-1)# owner
device(config-if-e1000-1/6-vrid-1)# ip-address 10.53.5.1
device(config-if-e1000-1/6-vrid-1)# activate
VRRP router 1 for this interface is activating
```

Enabling a backup VRRP device

This task is performed on a device that is to be designated as a backup VRRP device. For example, Router 2 in Figure 85 on page 633 is assigned as a backup device. Repeat this task for all devices that are to be designated as backup devices.

1. On the device designated as a backup VRRP device, and from privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command, using the RBridge ID (which has an asterisk next to it when you run a **do show vcs** command).

```
device(config)# rbridge-id 102
```

3. Globally enable VRRP.

```
device(config-rbridge-id-102)# protocol vrrp
```

```
device(config)# protocol vrrp
```

4. Configure the Ethernet interface for Router 2.

```
device(config-rbridge-id-102)# interface tengigabitethernet 102/1/5
```

```
device(config)# interface ethernet 1/5
```

```
device(config)# interface ethernet 0/5
```

5. Configure the IP address of interface:

```
device(config-if-te-102/1/5)# ip address 192.168.4.3/24
```

```
device(config-if-eth-1/5)# ip address 192.168.4.3/24
```

```
device(conf-if-eth-0/5)# ip address 192.168.4.3/24
```

NOTE
This router will become the backup router to Router 1.

6. Assign Router 2 to the same VRRP group as Router 1.

```
device(config-if-te-102/1/5)# vrrp-group 1
device(config-if-eth-1/5)# vrrp-group 1
device(conf-if-eth-0/5)# vrrp-group 1
```

7. To assign Group 1 a virtual IP address, use the same virtual IP address you used for Router 1.

```
device(config-vrrp-group-1)# virtual-ip 192.168.4.1
```

The following example configures a backup VRRP device.

```
device# configure
device(config)# rbridge-id 102
device(config-rbridge-id-102)# protocol vrrp
device(config-rbridge-id-102)# interface tengigabitethernet 102/1/5
device(config-if-te-102/1/5)# ip address 192.168.4.3/24
device(config-if-te-102/1/5)# vrrp-group 1
device(config-vrrp-group-1)# virtual-ip 192.168.4.1

device# configure
device(config)# protocol vrrp
device(config)# interface ethernet 1/5
device(config-if-eth-1/5)# ip address 192.168.4.3/24
device(config-if-eth-1/5)# vrrp-group 1
device(config-vrrp-group-1)# virtual-ip 192.168.4.1

device# configure
device(config)# protocol vrrp
device(config)# interface ethernet 0/5
device(conf-if-eth-0/5)# ip address 192.168.4.3/24
device(conf-if-eth-0/5)# vrrp-group 1
device(config-vrrp-group-1)# virtual-ip 192.168.4.1
```

Enabling a backup VRRP device

This task is performed on any device that is designated as a backup VRRP device. For each VRRP virtual routing instance, there is one master device and all other devices are backups. For example, Router 2 in Figure 86 on page 637 is assigned as a backup device. Repeat this task for all devices that are to be designated as backup devices.

1. On the device designated as a backup VRRP device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP.

```
device(config)# router vrrp
```

3. Configure the Ethernet interface link.

```
device(config)# interface ethernet 1/5
```

4. Configure the IP address of the interface for Router 2. All devices configured for the same virtual router ID (VRID) must be on the same subnet.

```
device(config-if-e1000-1/5)# ip address 10.53.5.3/24
```

5. Assign Router 2 to VRID 1, the same VRID as Router 1.

```
device(config-if-e1000-1/5)# ip vrrp vrid 1
```

NOTE
You can assign a VRID number in the range of 1 through 255.

6. Designate this router as a backup VRRP device.

```
device(config-if-e1000-1/5-vrid-1)# backup priority 110
```

While configuring a backup device, you can set a priority that is used when a master VRRP device goes offline. The backup device with the highest priority will assume the role of master device.

7. Configure the number of seconds between hello messages.

```
device(config-if-e1000-1/5-vrid-1)# hello-interval 10
```

8. By default, backup VRRP devices do not send hello messages to advertise themselves to the master. Use the following command to enable a backup router to send hello messages to the master VRRP device.

```
device(config-if-e1000-1/5-vrid-1)# advertise backup
```

9. Configure the IP address of the VRID.

```
device(config-if-e1000-1/5-vrid-1)# ip-address 10.53.5.1
```

The VRID IP address is the same virtual IP address you used for Router 1.

10. Enable the VRRP session.

```
device(config-if-e1000-1/5-vrid-1)# activate
VRRP router 1 for this interface is activating
```

The following example configures a VRRP backup device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/5
device(config-if-e1000-1/5)# ip address 10.53.5.3/24
device(config-if-e1000-1/5)# ip vrrp vrid 1
device(config-if-e1000-1/5-vrid-1)# backup priority 110
device(config-if-e1000-1/5-vrid-1)# hello-interval 10
device(config-if-e1000-1/5-vrid-1)# advertise backup
device(config-if-e1000-1/5-vrid-1)# ip-address 10.53.5.1
device(config-if-e1000-1/5-vrid-1)# activate
VRRP router 1 for this interface is activating
```

Configuring simple text authentication on VRRP interfaces

A simple text password can be used for interface authentication in a network. VRRP uses the authentication type associated with the interfaces on which you define the virtual router ID (VRID).

A VRRP session must be configured and running.

If you configure your device interfaces to use a simple password to authenticate traffic, VRRP interfaces can be configured with the same simple password, and VRRP packets that do not contain the password are dropped. If your interfaces do not use authentication, neither does VRRP. Repeat this task on all interfaces on all devices that support the VRID.

NOTE
This task supports VRRPv2 and VRRP-Ev2 only. VRRPv3 and VRRP-Ev3 are not supported.

1. From privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP.

```
device(config)# router vrrp
```

3. Configure an Ethernet interface.

```
device(config)# interface ethernet 1/6
```

4. Enter the simple text password configuration using the **ip vrrp auth-type** command with a text password.

```
device(config-if-e10000-1/6)# ip vrrp auth-type simple-text-auth yourpwd
```

5. Verify the password on the interface using the **show ip vrrp** command with either the VRID or Ethernet options.

```
device(config-if-e10000-1/6-vrid-1)# show ip vrrp

Total number of VRRP routers defined: 1
Interface ethernet 1/6
auth-type simple text authentication
VRID 1
state backup
administrative-status enabled
mode owner
priority 99
current priority 99
hello-interval 1 sec
ip-address 10.53.5.1
backup routers 10.53.5.2
```

In this example, the authentication type is simple text authentication. A **show running-config** command with appropriate parameters will actually display the password. The output verifies the type of authentication.

Configuring MD5 authentication on VRRP interfaces

Interfaces can be configured with an MD5 encrypted password for authentication, and VRRP can use the same authentication type associated with the interfaces on which you define the virtual router ID (VRID).

If you configure your device interfaces to use an MD5 encrypted password to authenticate traffic, VRRP interfaces can be configured with the same MD5 password, and VRRP packets that do not contain the password are dropped. If your interfaces do not use authentication, neither does VRRP. Repeat this task on all interfaces on all devices that support the VRID.

1. From privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP.

```
device(config)# router vrrp
```

3. Specify an interface associated with the VRRP VRID.

```
device(config)# interface ethernet 1/6
```

4. Enter the MD5 password configuration using the **ip vrrp auth-type** command with a text password. The password will be encrypted when saved in the configuration file. When an MD5 authentication password is configured on an interface, a syslog message is displayed.

```
device(config-if-e10000-1/6)# ip vrrp auth-type md5-auth gy42mb
Aug 10 18:17:39 VRRP: Configuration VRRP_CONFIG_MD5_AUTHENTICATION request received
Aug 10 18:17:39 VRRP: Port 1/6, VRID 2 - send advertisement
Ver:3 Type:1 Vrid:2 Pri:240 #IP:1 AuthType:2 Adv:1 Chksum:0x0000
HMAC-MD5 CODE:[00000000000000000000400010]
IpAddr: 10.53.5.1
```

5. Verify the password on the interface using the **show ip vrrp** command.

```
device(config-if-e10000-1/6-vrid-1)# show ip vrrp

Total number of VRRP routers defined: 1
Interface ethernet 1/6
auth-type MD5 authentication
VRID 1
state backup
administrative-status enabled
mode owner
priority 99
current priority 99
hello-interval 1 sec
ip-address 10.53.5.1
backup routers 10.53.5.2
```

In this example, the auth-type is MD5 authentication where the entered password is encrypted. A **show run** command with appropriate parameters will actually display the encrypted password, and you can use the **enable password-display** command to actually display the encrypted password. The output verifies the type of authentication.

The following example enables MD5 authentication on Ethernet interface 1/6 and verifies the authentication type.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/6
device(config-if-e10000-1/6)# ip vrrp auth-type MD5 yourpwd
device(config-if-e10000-1/6-vrid-1)# show ip vrrp

Total number of VRRP routers defined: 1
Interface ethernet 1/6
auth-type MD5 authentication
VRID 1
state backup
administrative-status enabled
mode owner
priority 99
current priority 99
hello-interval 1 sec
ip-address 10.53.5.1
backup routers 10.53.5.2
```

Abdicating VRRP master device status

Changing the priority of a VRRP master device allows a temporary abdication of the master device status to allow a backup device with a higher priority to assume the master device role.

A VRRP session must be configured and running.

When you change the priority of a VRRP owner, the change takes effect only for the current power cycle. The change is not saved to the startup configuration file when you save the configuration, and it is not retained across a reload or reboot. Following a reload or reboot, the VRRP owner again has priority 255.

NOTE
This task is supports IPv4 VRRP only. IPv6 VRRP, VRRP-E, and IPv6 VRRP-E are not supported.

1. On the master device and from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP.

```
device(config)# router vrrp
```

3. Configure an Ethernet interface.

```
device(config)# interface ethernet 1/6
```

4. Enter the virtual router ID (VRID) for which the device is the VRRP owner.

```
device(config-if-e1000-1/6)# ip vrrp vrid 1
```

NOTE
You can assign a VRID number in the range of 1 through 255.

5. Enter a priority for this device that is lower than the priority of at least one backup device associated with the VRID.

```
device(config-if-e1000-1/6-vrid-1)# owner priority 99
```

6. Verify the abdication of the master device using the **show ip vrrp** command.

```
device(config-if-e10000-1/6-vrid-1)# show ip vrrp

Total number of VRRP routers defined: 1
Interface ethernet 1/6
auth-type no authentication
VRID 1
state backup
administrative-status enabled
mode owner
priority 99
current priority 99
hello-interval 1 sec
ip-address 10.53.5.1
backup routers 10.53.5.2
```

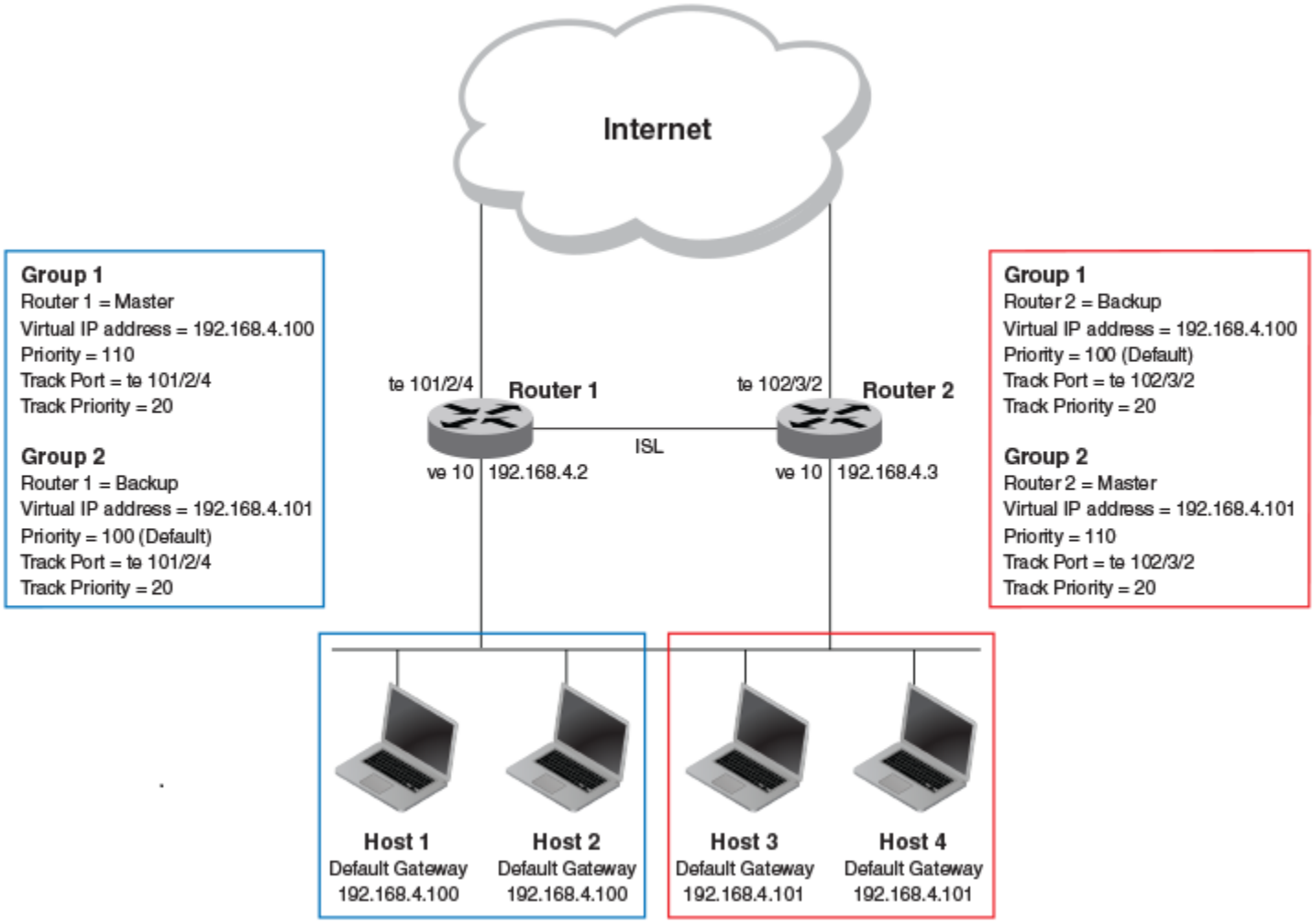
In this example, the mode shows this device as the owner of the virtual router (mode owner), but the VRRP priority for the device is only 99 and the state is now backup instead of master. The administrative status is still enabled. The output verifies that this device is now a backup device.

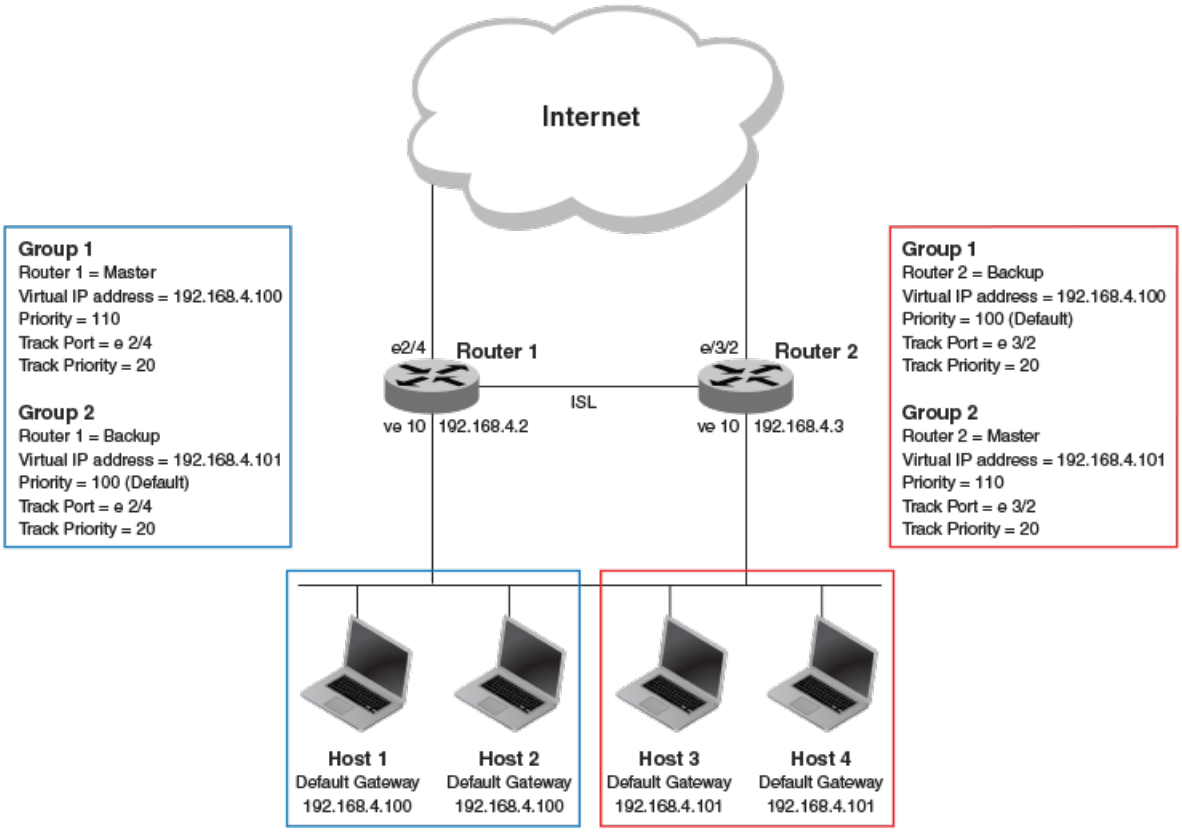
VRRP multigroup clusters

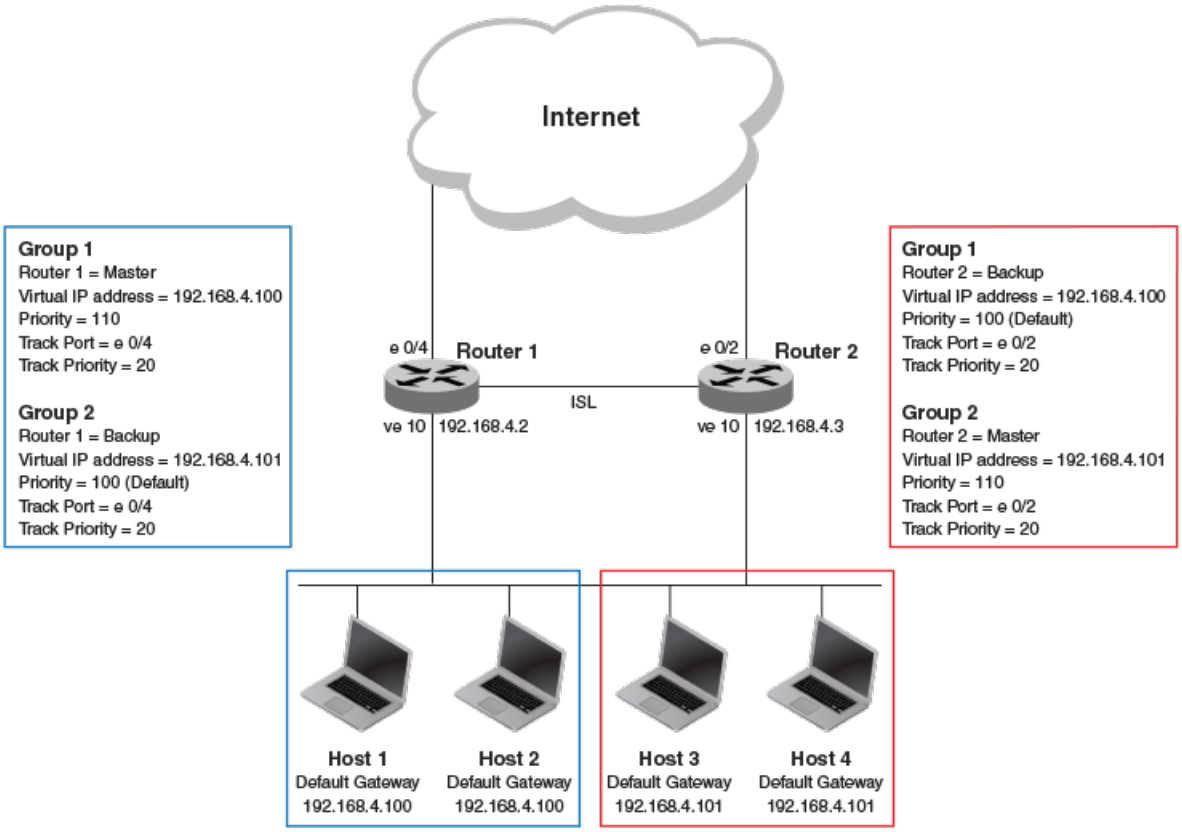
Multigroup clusters allow redundancy for host devices and are supported by both VRRP and VRRP-E version 2 and version 3.

The figure below depicts a commonly employed virtual router topology. This topology introduces redundancy by configuring two virtual router groups — the first group has Router 1 as the master and Router 2 as the backup, and the second group has Router 2 as the master and Router 1 as the backup. This type of configuration is sometimes called *Multigroup VRRP*.

FIGURE 87 Two routers configured for dual redundant network access for the host




Internet
Group 1
Router 1 = Master
Virtual IP address = 192.168.4.100
Priority = 110
Track Port = e 2/4
Track Priority = 20
Group 2
Router 1 = Backup
Virtual IP address = 192.168.4.101
Priority = 100 (Default)
Track Port = e 2/4
Track Priority = 20
e2/4
Router 1
ve 10
192.168.4.2
ISL
e/3/2
Router 2
ve 10
192.168.4.3
Group 1
Router 2 = Backup
Virtual IP address = 192.168.4.100
Priority = 100 (Default)
Track Port = e 3/2
Track Priority = 20
Group 2
Router 2 = Master
Virtual IP address = 192.168.4.101
Priority = 110
Track Port = e 3/2
Track Priority = 20
Host 1
Default Gateway
192.168.4.100
Host 2
Default Gateway
192.168.4.100
Host 3
Default Gateway
192.168.4.101
Host 4
Default Gateway
192.168.4.101



In this example, Router 1 and Router 2 use VRRP-E to load share as well as provide redundancy to the hosts. The load sharing is accomplished by creating two VRRP-E groups, each with its own virtual IP addresses. Half of the clients point to Group 1's virtual IP address as their default gateway, and the other half point to Group 2's virtual IP address as their default gateway. This enables some of the outbound Internet traffic to go through Router 1 and the rest to go through Router 2.

Router 1 is the master for Group 1 (master priority = 110) and Router 2 is the backup for Group 1 (backup priority = 100). Router 1 and Router 2 both track the uplinks to the Internet. If an uplink failure occurs on Router 1, its backup priority is decremented by 20 (track-port priority = 20) to 90, so that all traffic destined to the Internet is sent through Router 2 instead.

Similarly, Router 2 is the master for Group 2 (master priority = 110) and Router 1 is the backup for Group 2 (backup priority = 100). Router 1 and Router 2 are both tracking the uplinks to the Internet. If an uplink failure occurs on Router 2, its backup priority is decremented by 20 (track-port priority = 20) to 90, so that all traffic destined to the Internet is sent through Router 1 instead.

Configuring multigroup VRRP routing

Configuring VRRP multigroup clusters provides access redundancy to the host devices.

Before configuring this task, ensure that a virtual LAN, named vlan 10, has been created.

To implement the configuration of VRRP multigroup clusters as shown in Figure 87 on page 644, configure one VRRP-E router to act as a master in the first virtual router group and as a backup in the second virtual group. Then configure the second VRRP-E router to act as a backup in the first virtual group and as a master in the second virtual group.

This example is for VRRP-E. There are minor syntax differences for VRRP, which you can determine by consulting the appropriate command reference. The task steps below are configured on Router 1 and there are three configuration examples at the end of the task showing how to configure Router 1 as a backup and Router 2 as a master and a backup VRRP-E device.

1. Enter the **rbridge-id** command in global configuration mode, using the RBridge ID (which has an asterisk next to it when you run a **do show vcs** command).

   ```
   device(config)# rbridge-id 101
   ```

2. Enable the VRRP-E protocol globally.

   ```
   device(config-rbridge-id-101)# protocol vrrp-extended
   device(config)# protocol vrrp-extended
   ```

3. Configure the virtual Ethernet (ve) interface link for Router 1.

   ```
   device(config-rbridge-id-101)# interface ve 10
   ```

   NOTE
   You must first create **interface vlan 10** in global configuration mode.

4. Create a VLAN.

   ```
   device(config)# vlan 10
   ```

5. Bind the VLAN to the virtual Ethernet (ve) interface link for Router 1.

   ```
   device(config-vlan 10)# router-interface ve 10
   ```

6. Enter virtual Ethernet (ve) interface 10.

   ```
   device(config-vlan 10)# interface ve 10
   ```

7. Configure the IP address of the ve link for Router 1.

   ```
   device(config-if-Ve-10)# ip address 192.168.4.2/24
   ```

8. To assign Router 1 to a VRRP-E group called Group 1, enter the command:

   ```
   device(config-if-Ve-10)# vrrp-extended-group 1
   ```

9. Configure the tengigabitethernet port 101/2/4 as the tracking port for the interface ve 10, with a track priority of 20.

   ```
   device(config-vrrp-extended-group-1)# track te 101/2/4 priority 20
   ```

10. Configure the ethernet port 2/4 as the tracking port for the interface ve 10, with a track priority of 20.

    ```
    device(config-vrrp-extended-group-1)# track ethernet 2/4 priority 20
    ```

11. Configure the ethernet port 0/4 as the tracking port for the interface ve 10, with a track priority of 20.

    ```
    device(config-vrrp-extended-group-1)# track ethernet 0/4 priority 20
    ```

12. Configure an IP address for the virtual router.

```
device(config-vrrp-extended-group-1)# virtual-ip 192.168.4.100
```

NOTE
(For VRRP-E only) The address you enter with the **virtual-ip** command cannot be the same as a real IP address configured on the interface.

13. To configure Router 1 as the master, set the priority to a value higher than the default (which is 100).

```
device(config-vrrp-group-1)# priority 110
```

Router 1 as backup

The following example configures Router 1 as a backup device for VRRP-E group 2 by configuring a priority (100) that is a lower value than the priority set for Router 2 in VRRP-E group 2.

```
device(config)# rbridge-id 101
device(config-rbridge-id-101)# protocol vrrp-extended
device(config-rbridge-id-101)# interface ve 10
device(config-Ve-10)# ip address 192.168.4.2/24
device(config-Ve-10)# vrrp-extended-group 2
device(config-vrrp-extended-group-1)# track te 101/2/4 priority 20
device(config-vrrp-extended-group-1)# virtual-ip 192.168.4.101
device(config-vrrp-group-1)# priority 100

device(config)# protocol vrrp-extended
device(config)# vlan 10
device(config-vlan-10)# router-interface ve 10
device(config-vlan-10)# interface ve 10
device(config-if-Ve-10)# ip address 192.168.4.2/24
device(config-if-Ve-10)# vrrp-extended-group 2
device(config-vrrp-extended-group-1)# track ethernet 2/4 priority 20
device(config-vrrp-extended-group-1)# virtual-ip 192.168.4.101
device(config-vrrp-group-1)# priority 100

device(config)# protocol vrrp-extended
device(config)# vlan 10
device(config-vlan-10)# router-interface ve 10
device(config-vlan-10)# interface ve 10
device(config-if-Ve-10)# ip address 192.168.4.2/24
device(config-if-Ve-10)# vrrp-extended-group 2
device(config-vrrp-extended-group-1)# track ethernet 0/4 priority 20
device(config-vrrp-extended-group-1)# virtual-ip 192.168.4.101
device(config-vrrp-group-1)# priority 100
```

Router 2 as master

The following example configures Router 2 as the master device for VRRP-E group 2 by configuring a priority (110) that is a higher value than the priority set for Router 1 in VRRP-E group 2.

```
device(config)# rbridge-id 102
device(config-rbridge-id-102)# protocol vrrp-extended
device(config-rbridge-id-102)# interface ve 10
device(config-Ve-10)# ip address 192.168.4.3/24
device(config-Ve-10)# vrrp-extended-group 2
device(config-vrrp-extended-group-2)# track te 101/2/4 priority 20
device(config-vrrp-extended-group-2)# virtual-ip 192.168.4.101
device(config-vrrp-group-1)# priority 110

device(config)# protocol vrrp-extended
device(config)# vlan 10
device(config-vlan-10)# router-interface ve 10
device(config-vlan-10)# interface ve 10
device(config-if-Ve-10)# ip address 192.168.4.3/24
device(config-if-Ve-10)# vrrp-extended-group 2
device(config-vrrp-extended-group-2)# track ethernet 2/4 priority 20
device(config-vrrp-extended-group-2)# virtual-ip 192.168.4.101
device(config-vrrp-group-1)# priority 110

device(config)# protocol vrrp-extended
device(config)# vlan 10
device(config-vlan-10)# router-interface ve 10
device(config-vlan-10)# interface ve 10
device(config-if-Ve-10)# ip address 192.168.4.3/24
device(config-if-Ve-10)# vrrp-extended-group 2
device(config-vrrp-extended-group-2)# track ethernet 0/4 priority 20
device(config-vrrp-extended-group-2)# virtual-ip 192.168.4.101
device(config-vrrp-group-1)# priority 110
```

Router 2 as backup

The following example configures Router 2 as a backup device for VRRP-E group 1 by configuring a priority (100) that is a lower value than the priority set for Router 1 in VRRP-E group 1.

```
device(config)# rbridge-id 102
device(config-rbridge-id-102)# protocol vrrp-extended
device(config-rbridge-id-102)# interface ve 10
device(config-Ve-10)# ip address 192.168.4.3/24
device(config-Ve-10)# vrrp-extended-group 1
device(config-vrrp-extended-group-1)# track te 101/2/4 priority 20
device(config-vrrp-extended-group-1)# virtual-ip 192.168.4.100
device(config-vrrp-group-1)# priority 100

device(config)# protocol vrrp-extended
device(config)# vlan 10
device(config-vlan-10)# router-interface ve 10
device(config-vlan-10)# interface ve 10
device(config-if-Ve-10)# ip address 192.168.4.3/24
device(config-if-Ve-10)# vrrp-extended-group 1
device(config-vrrp-extended-group-1)# track ethernet 2/4 priority 20
device(config-vrrp-extended-group-1)# virtual-ip 192.168.4.100
device(config-vrrp-group-1)# priority 100

device(config)# protocol vrrp-extended
device(config)# vlan 10
device(config-vlan-10)# router-interface ve 10
device(config-vlan-10)# interface ve 10
device(config-if-Ve-10)# ip address 192.168.4.3/24
device(config-if-Ve-10)# vrrp-extended-group 1
device(config-vrrp-extended-group-1)# track ethernet 0/4 priority 20
device(config-vrrp-extended-group-1)# virtual-ip 192.168.4.100
device(config-vrrp-group-1)# priority 100
```

Tracked ports and track priority with VRRP and VRRP-E

Port tracking allows interfaces not configured for VRRP or VRRP-E to be monitored for link-state changes that can result in dynamic changes to the VRRP device priority.

A tracked port allows you to monitor the state of the interfaces on the other end of a route path. A tracked interface also allows the virtual router to lower its priority if the exit path interface goes down, allowing another virtual router in the same VRRP (or VRRP-E) group to take over. When a tracked interface returns to an up state, the configured track priority is added to the current virtual router priority value. The following conditions and limitations exist for tracked ports:

- Track priorities must be lower than VRRP or VRRP-E priorities.
- The dynamic change of router priority can trigger a master device switchover if preemption is enabled. However, if the router is an owner, the master device switchover will not occur.
- The maximum number of interfaces that can be tracked for a virtual router is 16.
- Port tracking is allowed for physical interfaces and port channels.

Configuring VRRP port tracking

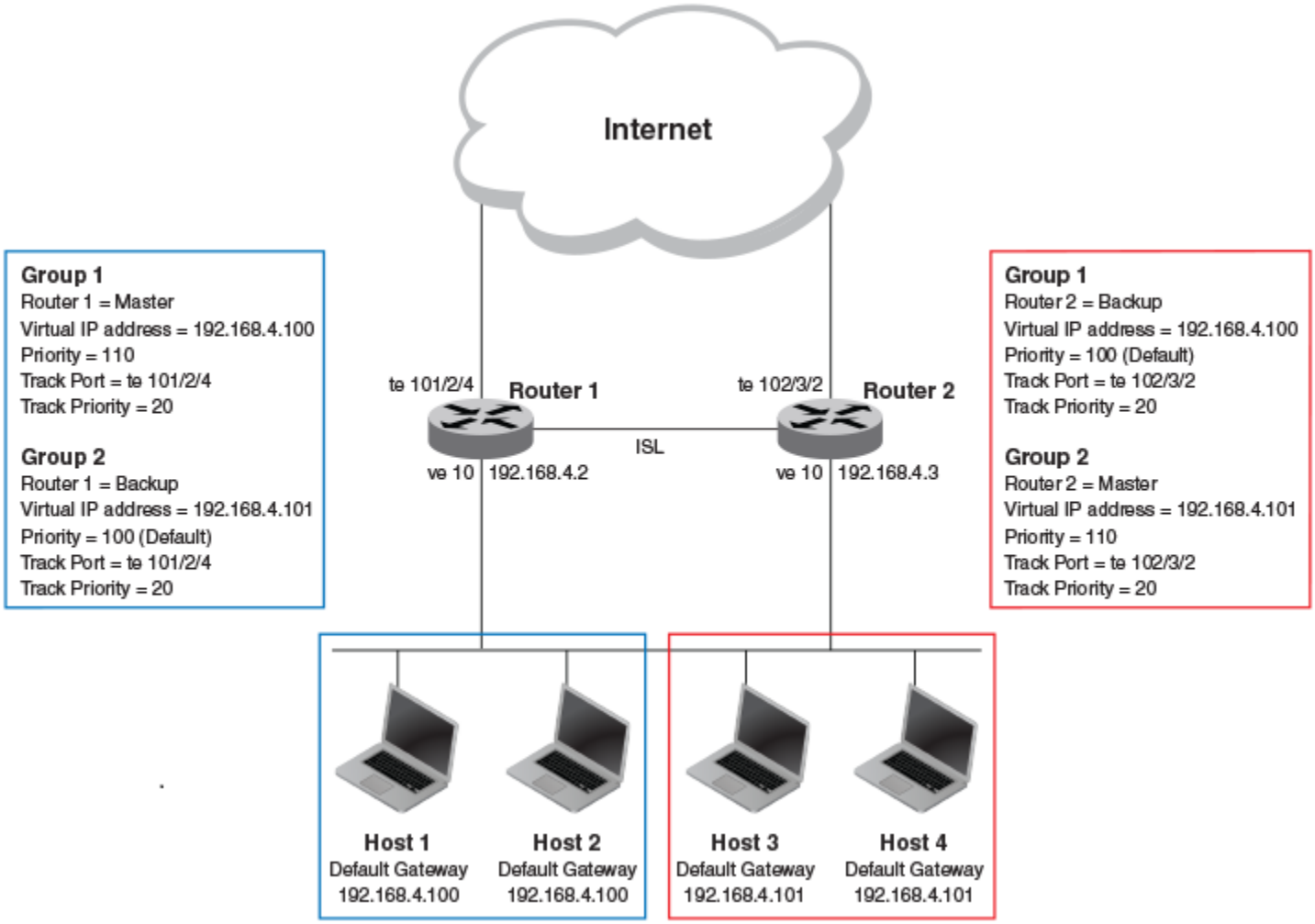
Configuring port tracking on an exit path interface and setting a priority on a VRRP device enables VRRP to monitor the interface. If the interface goes down, the device priority is lowered and another backup device with a higher priority assumes the role of master.

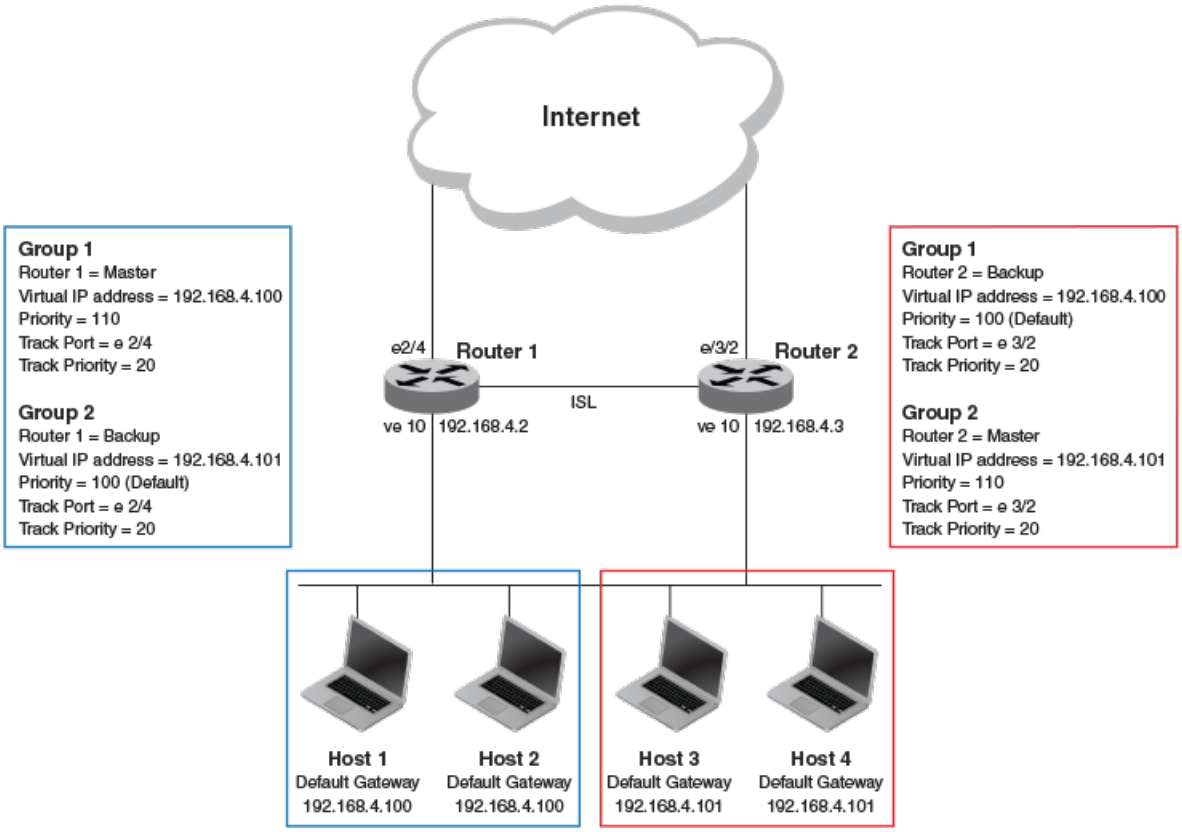
In the following task steps, interface tengigabitethernet 101/2/4 on Router 1 (shown in the diagram below) is configured to be tracked, and if the interface fails, the VRRP priority of Router 1 is lowered by a value of 20. Router 1 is the master VRRP device for group 1 and a lower priority triggers a backup device with a higher priority, Router 2, to become the new master for Group 1. Perform this task on the device on which the tracked interface exists.

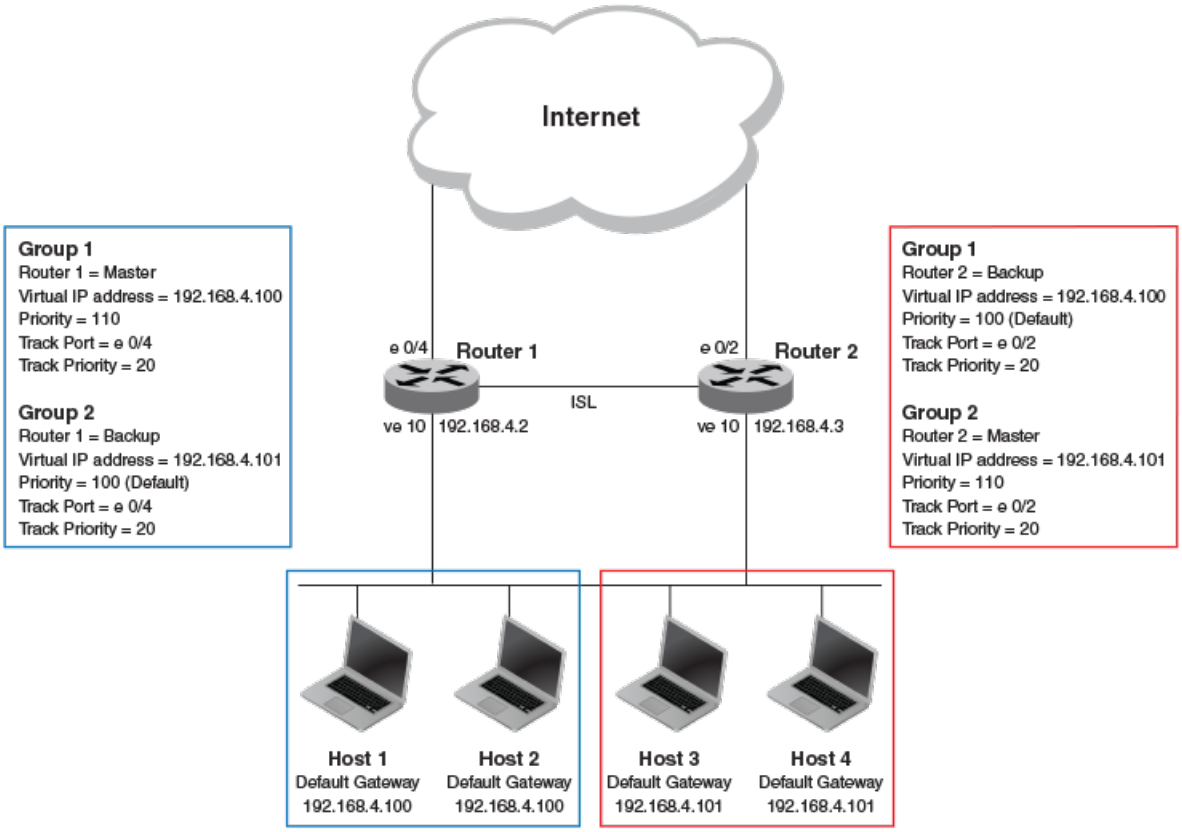
In the following task steps, interface ethernet 2/4 on Router 1 (shown in the diagram below) is configured to be tracked, and if the interface fails, the VRRP priority of Router 1 is lowered by a value of 20. Router 1 is the master VRRP device for group 1 and a lower priority triggers a backup device with a higher priority, Router 2, to become the new master for Group 1. Perform this task on the device on which the tracked interface exists.

In the following task steps, interface ethernet 0/4 on Router 1 (shown in the diagram below) is configured to be tracked, and if the interface fails, the VRRP priority of Router 1 is lowered by a value of 20. Router 1 is the master VRRP device for group 1 and a lower priority triggers a backup device with a higher priority, Router 2, to become the new master for Group 1. Perform this task on the device on which the tracked interface exists.

FIGURE 88 Multigroup VRRP routing topology




Internet
Group 1
Router 1 = Master
Virtual IP address = 192.168.4.100
Priority = 110
Track Port = e 2/4
Track Priority = 20
Group 2
Router 1 = Backup
Virtual IP address = 192.168.4.101
Priority = 100 (Default)
Track Port = e 2/4
Track Priority = 20
e2/4
Router 1
ve 10
192.168.4.2
ISL
e/3/2
Router 2
ve 10
192.168.4.3
Group 1
Router 2 = Backup
Virtual IP address = 192.168.4.100
Priority = 100 (Default)
Track Port = e 3/2
Track Priority = 20
Group 2
Router 2 = Master
Virtual IP address = 192.168.4.101
Priority = 110
Track Port = e 3/2
Track Priority = 20
Host 1
Default Gateway
192.168.4.100
Host 2
Default Gateway
192.168.4.100
Host 3
Default Gateway
192.168.4.101
Host 4
Default Gateway
192.168.4.101



1. Enter the **rbridge-id** command in global configuration mode, using the RBridge ID (which has an asterisk next to it when you run a **do show vcs** command).

```
device(config)# rbridge-id 101
```

2. Enable VRRP globally.

```
device(config-rbridge-id-101)# protocol vrrp
device(config)# protocol vrrp
```

3. Enter interface configuration mode and run the following command:

```
device(config-rbridge-id-101)# interface ve 10
device(config)# interface ve 10
```

4. Run the following command to enter group configuration mode.

```
device(config-if-Ve-10)# vrrp-group 1
```

5. Enter the **track** command to set the track port and priority:

```
device(config-vrrp-group-1)# track te 101/2/4 priority 20
device(config-vrrp-group-1)# track ethernet 2/4 priority 20
device(config-vrrp-group-1)# track ethernet 0/4 priority 20
```

The following example shows how to configure an Ethernet interface on Router 2 to be tracked, and if the interface fails, the VRRP priority of Router 2 is lowered by a value of 40. Router 2 is the master VRRP device for group 2 and a lower priority triggers a backup device, Router 1, to become the new master for group 2.

```
device(config)# rbridge-id 102
device(config-rbridge-id-102)# protocol vrrp
device(config-rbridge-id-102)# interface ve 10
device(config-Ve-10)# vrrp-group 2
device(config-vrrp-group-1)# track te 101/2/4 priority 20

device(config)# protocol vrrp
device(config)# interface ve 10
device(config-if-Ve-10)# vrrp-group 2
device(config-vrrp-group-1)# track ethernet 2/4 priority 20

device(config)# protocol vrrp
device(config)# interface ve 10
device(config-if-Ve-10)# vrrp-group 2
device(config-vrrp-group-1)# track ethernet 0/4 priority 20
```

Tracking ports and setting the VRRP priority

Configuring port tracking on an exit path interface and setting a priority on a VRRP device enables VRRP to monitor the interface. For VRRP, if the interface goes down, the device priority is set to the priority value and another backup device with a higher priority assumes the role of master. For VRRP-E, if the interface goes down, the device priority is lowered by the priority value and another backup device with a higher priority assumes the role of master.

Configure this task on the device on which the tracked interface exists.

NOTE
Only IPv4/IPv6 IPsec tunnels can be configured with a specific **track-port** command priority. Other interfaces use the track priority associated with the **owner** or **backup** commands.

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

   ```
   device# configure terminal
   ```

2. Enter the **router vrrp** command to configure VRRP globally.

   ```
   device(config)# router vrrp
   ```

3. Configure the Ethernet interface.

   ```
   device(config)# interface ethernet 1/6
   ```

4. Enter the IP address for the interface to be used for the virtual router ID (VRID).

   ```
   device(config-if-e10000-1/6)# ip address 10.53.5.3/24
   ```

5. Enter the following command to enter the appropriate VRRP virtual router ID (VRID) mode.

   ```
   device(config-if-e10000-1/6)# ip vrrp vrid 1
   ```

6. Enter the **track-port** command to set the track port and priority:

   ```
   device(config-if-e10000-1/6-vrid-1)# track-port tunnel 1 priority 40
   ```

   The priority value is used when a tracked port goes down and the new priority is set to this value. Ensure that the priority value is lower than the priority set for any existing master or backup device to force a renegotiation for the master device.

The following example shows how to configure tunnel 1 on virtual router 1 to be tracked; if the interface fails, the VRRP priority of the device becomes 40, forcing a negotiation for a new master device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/6
device(config-if-e10000-1/6)# ip address 10.53.5.1/24
device(config-if-e10000-1/6)# ip vrrp vrid 1
device(config-if-e10000-1/6-vrid-1)# track-port tunnel 1 priority 40
```

VRRP owner preemption and interface tracking

Owner preemption can be enabled for a VRRP session by setting a priority value less than the default value of 255. Tracking interfaces for an VRRP owner session can be enabled by setting a tracked interface priority value.

In VRRP, an IP address is assigned to a virtual router and owner device is the physical router whose real interface IP address is the IP address that you assign to the virtual router. A virtual router is a collection of physical routers that can use VRRP to provide redundancy to routers within a LAN. The owner device responds to packets addressed to any of the IP addresses in the corresponding virtual router. By default, the owner is the master VRRP device and has the highest priority of 255.

VRRP owner preemption allows a lower device priority to be set and if a backup VRRP device has a higher priority, the backup device assumes the master VRRP role and the current owner device becomes a backup device. One use case for VRRP owner preemption is when you need to install a software upgrade and the device must be reloaded. If the device remains the master device during the upgrade and reload process, network data may be lost. After VRRP preemption is disabled, the owner priority reverts to 255 and the owner device assumes the role of the master VRRP device again.

The tracking of interfaces on a VRRP owner session can be enabled, regardless of whether owner preemption is enabled or disabled. The tracked interface value configured for the owner device overrides any configured individual tracked interface priorities. When owner preemption is enabled, the current priority of the owner session is determined by the configured owner priority and the track priorities of tracked ports. If a backup VRRP device has a higher priority than the combination of the owner and tracked interface priority, it assumes the role of master VRRP device. When owner preemption is disabled, the tracking of interfaces does not affect the current owner priority.

Enabling VRRP owner preemption and interface tracking

VRRP owner preemption is enabled when the owner priority is set to a value less than the default value of 255. Interfaces can be tracked on owner devices when the interface track priority is configured.

This task assumes that VRRP master and backup devices are already configured.

VRRP preemption can be useful when a software upgrade must be installed and reloaded on the owner device. Interface tracking is a useful troubleshooting tool to manage VRRP master and backup device priorities when interfaces fail. Reducing the priority value of the VRRP owner device may cause a backup device to assume the role of the master device.

1. On the device designated as the master VRRP device, and from privileged EXEC mode, enter configuration mode.

   ```
   device# configure terminal
   ```

2. Globally enable VRRP.

   ```
   device(config)# protocol vrrp
   ```

3. Configure the Ethernet interface link for Router 1.

   ```
   device(config)# interface ethernet 0/6
   ```

4. Configure the IP address of the interface.

```
device(conf-if-eth-0/6)# ip address 192.168.4.1/24
```

5. Assign the device to a VRRP group named 1.

```
device(conf-if-eth-0/6)# vrrp-group 1
```

6. Assign a virtual router IP address.

```
device(config-vrrp-group-1)# virtual-ip 192.168.4.1
```

   NOTE
   For VRRP, the physical router whose IP address is the same as the virtual router group IP address becomes the owner and master.

7. Enable owner preemption by assigning a lower priority than the master default of 255.

```
device(config-vrrp-group-1)# owner priority 200
```

8. Assign an interface track priority for the owner session. This value overrides any configured individual tracked interface priorities.

```
device(config-vrrp-group-1)# owner track-priority 20
```

   NOTE
   When owner preemption is enabled, if a backup VRRP device has a higher priority than the combination of the owner and tracked interface priority, it assumes the role of the master VRRP device.

The following example configures the VRRP owner device priority to 200 and the tracked interface priority to 20.

```
device# configure terminal
device(config)# protocol vrrp
device(config)# interface ethernet 0/6
device(conf-if-eth-0/6)# ip address 192.168.4.1/24
device(conf-if-eth-0/6)# vrrp-group 1
device(config-vrrp-group-1)# virtual-ip 192.168.4.1
device(config-vrrp-group-1)# owner priority 200
device(config-vrrp-group-1)# owner track-priority 20
```

VRRP backup preemption

Preemption of a backup VRRP device acting as a master device is allowed when another backup device has a higher priority.

By default, preemption is enabled for VRRP. In VRRP, preemption allows a backup device with the highest priority to become the master device when the master (also the owner) device goes offline. If another backup device is added with a higher priority, it will assume the role of the master VRRP device. In some larger networks there may be a number of backup devices with varying levels of priority, and preemption can cause network flapping. To prevent the flapping, disable preemption.

NOTE
If preemption is disabled for VRRP, the owner device is not affected because the owner device always preempts the active master. When the owner device is online, the owner device assumes the role of the master device regardless of the setting for the preempt parameter.

In VRRP-E, preemption is disabled by default. In situations where a new backup device is to be added with a higher priority, preemption can be enabled. There are no owner devices in VRRP-E to automatically preempt a master device.

Enabling VRRP backup preemption

Allowing a backup VRRP device that is acting as the master to be preempted by another backup device with a higher priority value.

A VRRP session must be globally enabled using the **protocol vrrp** command in RBridge ID configuration mode.

A VRRP session must be globally enabled using the **protocol vrrp** command in global configuration mode.

Preemption is enabled by default for VRRP, and disabled by default on VRRP-E. Assuming that preemption is disabled in a VRRP session, perform the following steps on a VRRP backup device.

1. In RBridge ID configuration mode, configure the tengigabitethernet interface for the device.

   ```
   device(config-rbridge-id-102)# interface tengigabitethernet 102/1/5
   ```

2. From global configuration mode, configure the ethernet interface for the device.

   ```
   device(config)# interface ethernet 1/5
   device(config)# interface ethernet 0/5
   ```

3. Configure the IP address of the interface:

   ```
   device(conf-if-te-102/1/5)# ip address 192.168.4.3/24
   device(conf-if-eth-1/5)# ip address 192.168.4.3/24
   device(conf-if-eth-0/5)# ip address 192.168.4.3/24
   ```

   NOTE
   This router is a backup router.

4. Assign the device to VRRP group 1.

   ```
   device(conf-if-te-102/1/5)# vrrp-group 1
   device(conf-if-eth-1/5)# vrrp-group 1
   device(conf-if-eth-0/5)# vrrp-group 1
   ```

5. Enter the **preempt-mode** command to configure backup preemption.

   ```
   device(conf-vrrp-group-1)# preempt-mode
   ```

   If a backup device has a higher priority than the current master device, the backup device will assume the role of the VRRP master device after preemption is enabled.

The following example enables preemption on a backup VRRP device.

```
device(config-rbridge-id-101)# interface tengigabitethernet 102/1/5
device(conf-if-te-101/1/5)# ip address 192.168.4.3/24
device(conf-if-te-101/1/5)# vrrp-group 1
device(config-vrrp-group-1)# preempt-mode

device(config)# interface ethernet 1/5
device(conf-if-eth-1/5)# ip address 192.168.4.3/24
device(conf-if-eth-1/5)# vrrp-group 1
device(config-vrrp-group-1)# preempt-mode

device(config)# interface ethernet 0/5
device(conf-if-eth-0/5)# ip address 192.168.4.3/24
device(conf-if-eth-0/5)# vrrp-group 1
device(config-vrrp-group-1)# preempt-mode
```

Disabling VRRP backup preemption

VRRP backup preemption can be disabled to avoid route flapping when a backup VRRP device that is acting as the master device could be preempted by another backup device with a higher priority value.

A VRRP or VRRP-E session must be globally enabled using the **router vrrp** or **router vrrp-extended** command in global configuration mode.

Preemption is enabled by default for VRRP and VRRP-E, but if several devices come back online with higher priorities than the original backup device, route flapping can occur as these devices preempt each other. The following steps can be used when you want to avoid a backup device acting as the master from being preempted by another backup device with a higher priority value.

1. Enter interface configuration mode.

   ```
   device(config)# interface ethernet 1/5
   ```

2. Enter the IP address for the interface to be used for the virtual router ID (VRID).

   ```
   device(config-if-e10000-1/5)# ip address 10.53.5.3/24
   ```

3. Enter the following command to enter the appropriate VRRP VRID mode.

   ```
   device(config-if-e10000-1/5)# ip vrrp vrid 1
   ```

4. Enter the **non-preempt-mode** command to disable backup preemption.

   ```
   device(config-if-e10000-1/5-vrid-1)# non-preempt-mode
   ```

   Even if a backup device has a higher priority than the current backup acting as a master device, the backup device will not assume the role of the VRRP master device.

The following example disables preemption on a backup VRRP device.

```
device(config)# router vrrp
device(config)# interface ethernet 1/5
device(config-if-e10000-1/5)# ip address 10.53.5.3/24
device(config-if-e10000-1/5)# ip vrrp vrid 1
device(config-if-e10000-1/5-vrid-1)# non-preempt-mode
```

Accept mode for backup VRRP devices

Accept mode allows a backup VRRP device to respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

For each VRRP virtual routing instance, there is one master device and all other devices are backups. Accept mode allows some network management functionality for backup VRRP devices, providing the ability to respond to ping, traceroute, and Telnet packets. Troubleshooting network connections to the VRRP nonowner master device is difficult unless accept mode is enabled. By default, accept mode is enabled and non-owner VRRP devices will accept packets destined for the IPv4 or IPv6 VRID addresses if they become master.

> NOTE
> The accept mode functionality enables a VRRP nonowner master device to respond to ping, Telnet, and traceroute packets, but the device will not respond to SSH packets.

Disabling accept mode on a backup VRRP device

When accept mode is disabled on a backup VRRP device, it will not respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

Perform this task on any device that is designated as a backup VRRP device. The functionality is activated if the backup device becomes a master VRRP device. Repeat this task for all devices that are to be designated as backup devices.

> NOTE
> The accept mode functionality does not support SSH packets.

1. On the device designated as a backup VRRP device, from privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Disable VRRP accept mode for all instances of VRRP on the device.

   ```
   device(config)# vrrp-acceptmode-disable
   ```

   You can re-enable accept mode by using the **no** form of this command.

Accept mode for backup VRRP devices

Accept mode allows a backup VRRP device to respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

For each VRRP virtual routing instance, there is one master device and all other devices are backups. Accept mode allows some network management functionality for backup VRRP devices, providing the ability to respond to ping, traceroute, and Telnet packets. Troubleshooting network connections to the VRRP non-owner master device is difficult unless accept mode is enabled. By default, accept mode is enabled and non-owner VRRP devices will accept packets destined for the IPv4 or IPv6 VRID addresses if they become master.

> NOTE
> The accept mode functionality enables a VRRP non-owner master device to respond to ping, Telnet, and traceroute packets, but the device will not respond to SSH packets.

Disabling accept mode on a backup VRRP device

When accept mode is disabled on a backup VRRP device, it will not respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

Perform this task on any device that is designated as a backup VRRP device. The functionality is activated if the backup device becomes a master VRRP device. Repeat this task for all devices that are to be designated as backup devices.

> NOTE
> The accept mode functionality does not support SSH packets.

1. On the device designated as a backup VRRP device, from privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Run the **rbridge-id** command with the RBridge ID.

   ```
   device(config)# rbridge-id 1
   ```

3. Disable VRRP accept mode for all instances of VRRP on the device.

```
device(config-rbridge-id-1)# vrrp-acceptmode-disable
```

You can use the **no** form of this command to re-enable accept mode.

Virtual router MAC address

When you configure a virtual routing ID (VRID), the software automatically uses the MAC address as the MAC address of the virtual router. The first five octets of the address are the standard MAC prefix for VRRP packets. The last octet is the VRID.

When the virtual router becomes the master router, it broadcasts a gratuitous ARP (GARP) request containing the virtual router's MAC address for each IP address associated with the virtual router. Hosts use the MAC address of the virtual router in routed traffic they send to their default IP gateway.

You can manually configure a unique virtual MAC address for each IPv4 and IPv6 VRRP instance per VRID. If there is no manually configured virtual MAC address for a VRRP instance, the system automatically assigns one.

The ability to configure a unique virtual MAC address is subject to the following limitations:

- This feature does not support configurable VRRP virtual MAC addresses over Multi-Chassis Trunking (MCT).
- This feature has no impact on short-path forwarding for VRRP-E.

NOTE
A virtual MAC address can be dynamically updated while a VRRP or VRRP-E session is enabled. When the VRRP or VRRP-E virtual MAC address is modified on the master device, expect a traffic drop until the host device receives the GARP or Router Advertisement (RA) containing the updated virtual MAC address from the master VRRP device.

Configuring unique virtual MAC addresses per VRID

In addition to system-configured standards-based virtual MAC addresses, you can manually configure a unique virtual MAC address for each IPv4 and IPv6 VRRP instance per virtual routing ID (VRID). If there is no manually configured virtual MAC address for a VRRP instance, the system automatically assigns one.

For the Brocade NetIron MLX Series platform, you can configure a maximum of 2000 virtual MAC addresses. For Brocade NetIron CES Series and Brocade NetIron CER Series platforms, you can configure a maximum of 255 virtual MAC addresses.

NOTE
System-assigned virtual MAC addresses and manually configured virtual MAC addresses can exist at the same time on the device under the same VRID, however the configured value takes precedence. When the configured value is deleted, the assigned value again applies.

NOTE
A virtual MAC address can be dynamically updated while a VRRP or VRRP-E session is enabled. When the VRRP or VRRP-E virtual MAC address is modified on the master device, expect a traffic drop until the host device receives the GARP request or Router Advertisement (RA) containing the updated virtual MAC address from the master VRRP device.

To configure a unique VRRP or VRRP-E virtual MAC address for a VRID, complete the following steps.

1. On the device designated as a VRRP-E device, from privileged EXEC mode, enter configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable the VRRP-E protocol.

```
device(config)# router vrrp-extended
```

3. Configure the ethernet interface link.

```
device(config-vrrpe-router)# interface ethernet 1/5
```

4. Configure the IP address of the interface. All devices configured for the same VRID must be on the same subnet.

```
device(conf-if-e1000-1/5)# ip address 10.53.5.3/24
```

5. Assign the device to VRID 1.

```
device(conf-if-e1000-1/5)# ip vrrp-extended vrid 1
```

NOTE
You can assign a VRID number in the range of 1 through 255.

6. To configure an IPv4 virtual MAC address for VRID 1 (for example), enter the following command at the configure VRID level of the CLI:

```
device(config-if-e1000-1/5-vrid-1)# virtual-mac aaaa.bbbb.cccc
```

NOTE
System-assigned virtual MAC addresses and manually configured virtual MAC addresses can exist at the same time on the device under the same VRID, however the configured value takes precedence. When the configured value is deleted, the assigned value again applies.

7. To display IPv4 VRRP-E virtual MAC address configuration information about VRID 1 (for example), enter the following command:

```
device# show ip vrrp-extended vrid 1

Interface 1/5
----------------
auth-type md5-authentication
VRID 1 (index 1)
interface 1/5
state master
administrative-status disabled
mode non-owner(backup)
virtual mac aaaa.bbbb.cccc (configured)
priority 100
current priority 100
track-priority 5
hello-interval 1 sec
backup hello-interval 60 sec
slow-start timer (configured) 30 sec
advertise backup disabled
dead-interval 0 ms
preempt-mode true
virtual ip address 10.20.1.100
short-path-forwarding disabled
```

The following example configures an IPv4 virtual MAC address for VRID 1 on a VRRP-E device. This example shows all the steps required to configure and activate a VRRP-E device.

```
device# configure terminal
device(config)# router vrrp-extended
device(config)# interface ethernet 1/5
device(config-if-e1000-1/5)# ip address 10.53.5.3/24
device(config-if-e1000-1/5)# ip vrrp-extended vrid 1
device(config-if-e1000-1/5-vrid-1)# backup priority 50 track-priority 10
device(config-if-e1000-1/5-vrid-1)# ip-address 10.53.5.1
device(config-if-e1000-1/5-vrid-1)# virtual-mac aaaa.bbbb.cccc
device(config-if-e1000-1/5-vrid-1)# activate
VRRP-E router 1 for this interface is activating
```

Configuring unique virtual MAC addresses per VRID

In addition to system-configured standards-based virtual MAC addresses, you can manually configure a unique virtual MAC address for each IPv4 and IPv6 VRRP or VRRP-E instance per virtual routing ID (VRID). If there is no manually configured virtual MAC address (VMAC) for a VRRP instance, the system automatically assigns one.

On SLX-OS devices, you can configure a maximum of 16 virtual router MAC addresses per device; this includes both IPv4 VRRP-E & IPv6 VRRP-E sessions.

> NOTE
> System-assigned virtual MAC addresses and manually configured virtual MAC addresses can exist at the same time on the device under the same VRID, however the configured value takes precedence. When the configured value is deleted, the assigned value again applies.

> NOTE
> A virtual MAC address can be dynamically updated while a VRRP or VRRP-E session is enabled. When the VRRP or VRRP-E virtual MAC address is modified on the master device, expect a traffic drop until the host device receives the GARP request or Router Advertisement (RA) containing the updated virtual MAC address from the master VRRP device.

To configure a unique VRRP or VRRP-E virtual MAC address for a VRID, complete the following steps.

1. On the device designated as a VRRP-E device, from privileged EXEC mode, enter configuration mode by issuing the **configure terminal** command.

   ```
   device# configure terminal
   ```

2. Globally enable the VRRP-E protocol.

   ```
   device(config)# protocol vrrp-extended
   ```

3. Configure a virtual ethernet interface link.

   ```
   device(config)# interface ve 10
   ```

4. Configure the IP address of the interface. All devices configured for the same VRID must be on the same subnet.

   ```
   device(config-if-Ve-10)# ip address 10.53.5.3/24
   ```

5. Assign the device to VRID group 12.

   ```
   device(config-if-Ve-10)# vrrp-extended-group 12
   ```

   > NOTE
   > You can assign a VRID number in the range of 1 through 16.

6. Manually configure an IPv4 virtual MAC address for virtual router group 12.

```
device(config-vrrp-extended-group-12)# virtual-mac 02e0.5200.0012
```

NOTE
System-assigned virtual MAC addresses and manually configured virtual MAC addresses can exist at the same time on the device under the same VRID, however the configured value takes precedence. When the configured value is deleted, the assigned value again applies.

7. To display IPv4 VRRP-E virtual MAC address configuration information about VRID 12 (for example), enter the following command:

```
device# show vrrp detail

Total number of VRRP session(s)   : 1

VRID 12
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRPE
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
  Authentication type: MD5 Authentication
  State: Initialize
  Session Master IP Address:
  Virtual IP(s): 192.168.4.100
  Virtual MAC Address: 02e0.5200.0112
.
.
.
```

The partial output shows the manually configured VMAC address.

The following example configures an IPv4 virtual MAC address for VRID 12 on a VRRP-E device.

```
device# configure terminal
device(config)# protocol vrrp-extended
device(config)# interface ve 10
device(config-if-Ve-10)# ip address 10.53.5.3/24
device(config-if-Ve-10)# vrrp-extended-group 12
device(config-vrrp-extended-group-12)# virtual-mac 02e0.5200.0012
```

Suppressing RIP route advertisements on VRRP backup devices

RIP route advertisement suppression can be enabled on VRRP backup devices to prevent other VRRP devices from learning multiple paths for a backed-up interface.

A VRRP or VRRP-E session with master and backup devices must be configured and running.

Normally, a VRRP or VRRP-E backup includes route information for the virtual IP address (the backed-up interface) in RIP advertisements. As a result, other devices receive multiple paths for the backed-up interface and might sometimes unsuccessfully use the path to the backup device rather than the path to the master device.

You can prevent the backups from advertising route information for the backed-up interface by enabling suppression of the advertisements.

NOTE
The command syntax is the same for VRRP and VRRP-E.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enable RIP.

```
device(config)# router rip
```

3. Suppress RIP route advertisements.

```
device(config-rip-router)# use-vrrp-path
```

The following example suppresses RIP advertisements for the backed-up interface.

```
device# configure terminal
device(config)# router rip
device(config-rip-router)# use-vrrp-path
```

VRRP-Ev2 overview

VRRP Extended (VRRP-E) is an extended version of VRRP. VRRP-E is designed to avoid the limitations in the standards-based VRRP.

VRRP-E is implemented the following differences from RFC 3768 which describes VRRPv2 to provide extended functionality and ease of configuration:

- VRRP-E does not include the concept of an owner device, and a master VRRP-E is determined by the priority configured on the device.
- While the VRRP-E virtual router IP address must belong in the same subnet as a real IP address assigned to a physical interface of the device on which VRRP-E is configured, it must not be the same as any of the actual IP addresses on any interface.
- Configuring VRRP-E uses the same task steps for all devices; there are no differences between master and backup device configuration. The device configured with the highest priority assumes the master role.

  NOTE
  VRRP-E is supported on the devices described in this guide. In a mixed-device environment, consult your documentation for the other devices to determine if VRRP-E is supported.
  VRRP-E does not interoperate with VRRP sessions.

Enabling a VRRP-E device

This task is performed on all devices that are designated as VRRP extended (VRRP-E) devices. While VRRP-E does not have owner devices, there is still a master device and backup devices with the master device determined by the device with the highest priority.

1. From privileged EXEC mode, enter configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Enter the **rbridge-id** command, using the RBridge ID (which has an asterisk next to it when you run a **do show vcs** command).

```
device(config)# rbridge-id 101
```

3. Globally enable VRRP-E.
   - Enter the **protocol vrrp** command on VDX 8770 devices.
   - Enter the **protocol vrrp-extended** command on VDX 6740 devices.

   ```
   device(config-rbridge-id-101)# protocol vrrp
   ```

   or

   ```
   device(config-rbridge-id-101)# protocol vrrp-extended
   ```

4. Globally enable VRRP-E.

   ```
   device(config)# protocol vrrp-extended
   ```

5. Configure the Virtual Ethernet (VE) interface link for the VRRP-E device.

   ```
   device(config-rbridge-id-101)# interface ve 10
   device(config)# interface ve 10
   ```

   Only ve interfaces are supported by VRRP-E.

6. Configure the IP address of the interface.

   ```
   device(conf-ve-10)# ip address 192.168.4.1/24
   device(config-if-Ve-10)# ip address 192.168.4.1/24
   ```

7. Assign the device to a group called Group 1.

   ```
   device(conf-ve-10)# vrrp-extended-group 1
   device(config-if-Ve-10)# vrrp-extended-group 1
   ```

   You can assign a group number in the range of 1 through 255.

8. Enter the **priority** command with a number to assign a priority.

   ```
   device(config-vrrp-group-1)# priority 110
   ```

   The VRRP-E device with the highest priority number becomes the master device.

9. Assign a virtual router IP address.

   ```
   device(config-vrrp-group-1)# virtual-ip 192.168.4.100
   ```

   NOTE
   For VRRP-E, the virtual router group IP address must not be the same as a real IP address configured on the interface.

Router 1

The following example configures a master VRRP-E device for group 1.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# protocol vrrp-extended
device(config-rbridge-id-101)# interface ve 10
device(conf-ve-10)# ip address 192.168.4.1/24
device(conf-ve-10)# vrrp-extended-group 1
device(config-vrrp-group-1)# priority 110
device(config-vrrp-group-1)# virtual-ip 192.168.4.100

device# configure terminal
device(config)# protocol vrrp-extended
device(config)# interface ve 10
device(config-if-Ve-10)# ip address 192.168.4.1/24
device(config-if-Ve-10)# vrrp-extended-group 1
device(config-vrrp-group-1)# priority 110
device(config-vrrp-group-1)# virtual-ip 192.168.4.100
```

Router 2

The following example configures a backup VRRP-E device for group 1. In the first configuration of VRRP-E for Router 1 the priority is set to 110, higher than the priority for Router 2 at 80. Router 1 assumes the role of the master VRRP-E device.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# protocol vrrp-extended
device(config-rbridge-id-101)# interface ve 10
device(config-ve-10)# ip address 192.168.4.3/24
device(config-ve-10)# vrrp-extended-group 1
device(config-vrrp-group-1)# priority 80
device(config-vrrp-group-1)# virtual-ip 192.168.4.100

device# configure terminal
device(config)# protocol vrrp-extended
device(config)# interface ve 10
device(config-if-Ve-10)# ip address 192.168.4.3/24
device(config-if-Ve-10)# vrrp-extended-group 1
device(config-vrrp-group-1)# priority 80
device(config-vrrp-group-1)# virtual-ip 192.168.4.100
```

Enabling a VRRP-E device

This task is performed on any device that is designated as a VRRP extended (VRRP-E) device. For each VRRP-E virtual routing instance, there is one master device and all other devices are backups; but, unlike VRRP, every device is configured as a backup and the device with the highest priority becomes the master VRRP-E device. Repeat this task for all devices that are to be designated as VRRP-E devices.

NOTE
Only VRRP or VRRP-E can be enabled in your network.

1. On the device designated as a VRRP-E device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP-E.

```
device(config)# router vrrp-extended
```

3. Configure the Ethernet interface link.

```
device(config-vrrpe-router)# interface ethernet 1/5
```

4. Configure the IP address of the interface. All devices configured for the same virtual router ID (VRID) must be on the same subnet.

```
device(config-if-e1000-1/5)# ip address 10.53.5.3/24
```

5. Assign the device to VRID 1.

```
device(config-if-e1000-1/5)# ip vrrp-extended vrid 1
```

   NOTE
   You can assign a VRID number in the range of 1 through 255.

6. Designate this router as a backup VRRP device.

```
device(config-if-e1000-1/5-vrid-1)# backup priority 50 track priority 10
```

   While configuring a backup device, you can set a priority that is used when a master VRRP device goes offline. The backup device with the highest priority will assume the role of master device.

7. Configure the IP address of the VRID.

```
device(config-if-e1000-1/5-vrid-1)# ip-address 10.53.5.254
```

   The IP address associated with the VRID must not be configured on any of the devices used for VRRP-E.

8. Enable the VRRP-E session.

```
device(config-if-e1000-1/5-vrid-1)# activate
VRRP-E router 1 for this interface is activating
```

The following example configures a VRRP-E device.

```
device# configure terminal
device(config)# router vrrp-extended
device(config-vrrpe-router)# interface ethernet 1/5
device(config-if-e1000-1/5)# ip address 10.53.5.3/24
device(config-if-e1000-1/5)# ip vrrp-extended vrid 1
device(config-if-e1000-1/5-vrid-1)# backup priority 50 track-priority 10
device(config-if-e1000-1/5-vrid-1)# ip-address 10.53.5.1
device(config-if-e1000-1/5-vrid-1)# activate
VRRP-E router 1 for this interface is activating
```

Configuring MD5 authentication on IPv4 VRRP-E interfaces

Interfaces can be configured with an MD5 encrypted password for authentication, and VRRP-E can use the same authentication type associated with the interfaces on which you define the virtual router.

VRRP Extended (VRRP-E) must be configured on the device and the interface associated with a virtual router group.

Any VRRP-E packets that do not contain the password are dropped. If your interfaces do not use authentication, neither does VRRP-E. Repeat this task on all interfaces on all devices that support the same virtual router group.

NOTE
VRRP-E is supported on the devices described in this guide. In a mixed-device environment, consult your documentation for the other devices to determine if VRRP-E is supported.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Globally enable VRRP-E.

```
device(config)# protocol vrrp-extended
```

3. Configure the Virtual Ethernet (VE) interface link for the VRRP-E device.

```
device(config)# interface ve 10
```

   Only ve interfaces are supported by VRRP-E.

4. Enter the MD5 password configuration using the **ip vrrp-extended auth-type** command with a text password. The password will be encrypted when saved in the configuration file.

```
device(config-if-Ve-10)# ip vrrp-extended auth-type md5-auth kfhb61qp
```

5. Exit to privileged EXEC mode.

```
device(config-if-Ve-10)# end
```

6. Display the VRRP-E configuration to verify that MD5 authentication is enabled.

```
device# show vrrp

Total number of VRRP session(s)   : 1

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRPE
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
  Authentication type: MD5 Authentication
  State: Initialize
  Session Master IP Address:
  Virtual IP(s): 192.168.4.100
  Configured Priority: 110 (default: 100); Current Priority: unset
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: DISABLE  (default: DISABLED)
  Advertise-backup: DISABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Disabled
  Revert Priority: unset; SPF reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========
  Statistics:
    Advertisements: Rx: 0, Tx: 0
    Gratuitous ARP: Tx: 0
```

The following example configures MD5 authentication for the specified VRRP-E interface.

```
device# configure terminal
device(config)# protocol vrrp-extended
device(config)# interface ve 10
device(config-if-Ve-10)# ip vrrp-extended auth-type md5-auth kfhb61qp
device(config-if-Ve-10)# end
device# show vrrp
```

Track routes and track priority with VRRP-E

Route tracking allows networks not configured for VRRP extended (VRRP-E) to be monitored for network reachability changes that can result in dynamic changes to the VRRP-E device priority.

Using network addresses, routes are tracked for online or offline events. The networks to be tracked can be either present or absent from the Routing Information Base (RIB). When route-tracking is enabled in the configured VRRP-E instance, the status of the tracked route is monitored. The priority of the VRRP-E device may be changed dynamically due to the following events:

- When a tracked route goes into an offline state, the configured track priority is subtracted from the current value of the VRRP-E device.
- When a tracked route returns to an online state, the configured track priority is added to the current value of the VRRP-E device.

NOTE
Network tracking is not supported by VRRP; only VRRP-E supports network tracking.

The dynamic change of device priority can trigger a switchover from a master VRRP-E device to a backup VRRP-E device if preemption is enabled.

Forward referencing for tracked routes is supported. The tracked route can be removed and added without the need to reconfigure the tracking for the route.

NOTE
Maximum number of routes that can be tracked for a virtual VRRP-E device is 16.

Configuring VRRP-E route tracking

Configuring route tracking on an exit path network and setting a priority on a VRRP Extended (VRRP-E) device enables VRRP-E to monitor the route. If the network goes down, the device priority is lowered and another backup device with a higher priority assumes the role of master.

In the following task steps, network 10.1.1.0/24 is configured to be tracked, and if the network goes offline, the VRRP priority of the current master device is lowered by a value of 20.

1. In global configuration mode, enter the **rbridge-id** command using the RBridge ID (which has an asterisk next to it when you run a **do show vcs** command).

   ```
   device(config)# rbridge-id 1
   ```

2. Enable VRRP-E globally.

   ```
   device(config-rbridge-id-1)# protocol vrrp-extended
   device(config)# protocol vrrp-extended
   ```

3. Enter interface configuration mode.

```
device(config-rbridge-id-1)# interface ve 100
device(config)# interface ve 100
```

4. Run the following command to enter group configuration mode.

```
device(config-if-Ve-100)# vrrp-extended-group 1
```

5. Enter the **track network** command to set the track network (route) and priority:

```
device(config-vrrp-group-1)# track network 10.1.1.0/24 priority 20
```

6. Return to privileged EXEC mode.

```
device(config-vrrp-group-1)# end
```

7. To view tracked networks with their priority and status, enter the following command:

```
device# show vrrp detail

================================================================================
Rbridge-id:1
================================================================================

Total number of VRRP session(s)   : 1

VRID 3
  Interface: Ve 100;  Ifindex: 1207959652
  Mode: VRRPE
.
.
.
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========

  Tracknetwork:
    Network(s)                 Priority Status
    =======                    ======== ===========
    10.1.1.0/24                20       Down

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 0, Tx: 0
    Neighbor Advertisements             : Tx: 0
.
.
.

device# show vrrp detail

Total number of VRRP session(s)   : 1

VRID 3
  Interface: Ve 100;  Ifindex: 1207959652
  Mode: VRRPE
.
.
.
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========

  Tracknetwork:
    Network(s)                 Priority Status
    =======                    ======== ===========
    10.1.1.0/24                20       Down

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 0, Tx: 0
```

```
    Neighbor Advertisements             : Tx: 0
.
.
.
```

The following example shows how to configure network 10.1.1.0/24 to be tracked. If the network goes down, the VRRP-E device priority is lowered by a value of 20. The lower priority may trigger a switchover and a backup device with a higher priority becomes the new master for VRRP-E group 1.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# protocol vrrp-extended
device(config-rbridge-id-1)# interface ve 100
device(conf-Ve-100)# vrrp-extended-group 1
device(config-vrrp-group-1)# track network 10.1.1.0/24 priority 20

device(config)# protocol vrrp-extended
device(config)# interface ve 100
device(config-if-Ve-100)# vrrp-extended-group 1
device(config-vrrp-group-1)# track network 10.1.1.0/24 priority 20
```

VRRP-E load-balancing using short-path forwarding

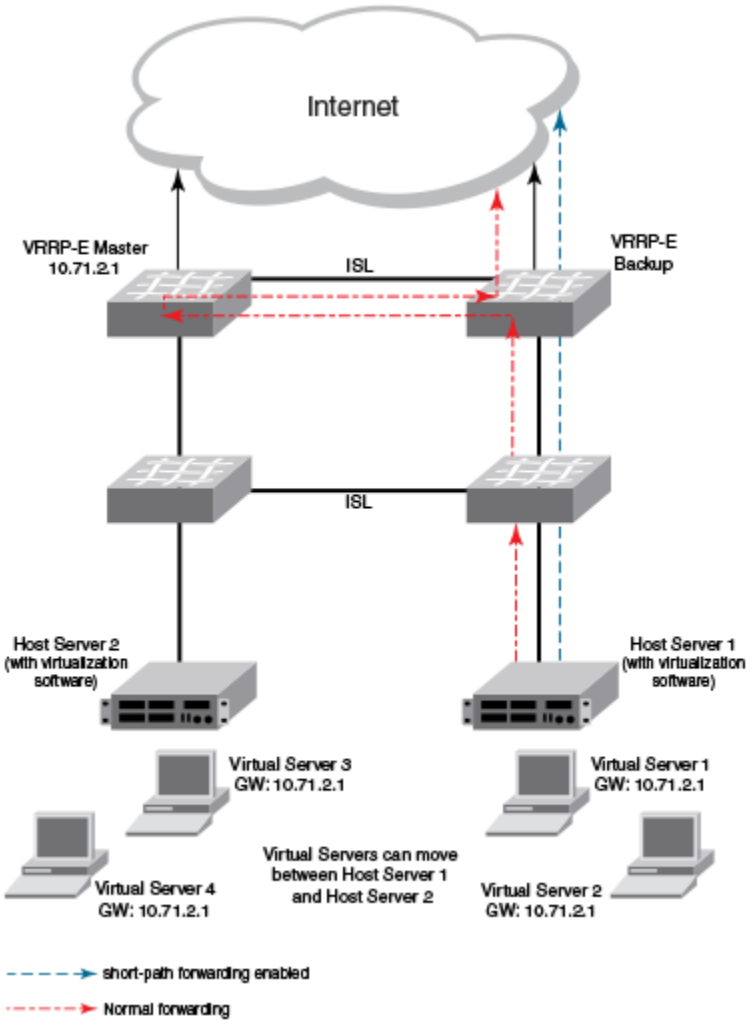
The VRRP-E Extension for Server Virtualization feature allows devices to bypass the VRRP-E master router and directly forward packets to their destination through interfaces on the VRRP-E backup router. This is called *short-path forwarding*. A backup router participates in a VRRP-E session only when short-path forwarding is enabled.

VRRP-E active-active load-balancing is available in VCS mode and uses flow-hashing techniques to determine the path. All nodes in the VCS are aware of all VRRP-E sessions and the participating RBridges in each session.

Packet routing with short-path forwarding to balance traffic load

When short-path forwarding is enabled, traffic load-balancing is performed because both master and backup devices can be used to forward packets.

FIGURE 89 Short-path forwarding



If you enable short-path forwarding in both master and backup VRRP-E devices, packets sent by Host Server 1 (in the figure) and destined for the Internet cloud through the device on which a VRRP-E backup interface exists can be routed directly to the VRRP-E backup device (blue dotted line) instead of being switched to the master router and then back (red dotted-dash line).

In the figure, load-balancing is achieved using short-path forwarding by dynamically moving the virtual servers between Host Server 1 and Host Server 2.

Short-path forwarding with revert priority

Revert priority is used to dynamically enable or disable VRRP-E short-path forwarding.

If short-path forwarding is configured with revert priority on a backup router, the revert priority represents a threshold for the current priority of the VRRP-E session. When the backup device priority is higher than the configured revert priority, the backup router is able to perform short-path forwarding. If the backup priority is lower than the revert priority, short-path forwarding is disabled.

Configuring VRRP-E load-balancing using short-path forwarding

VRRP-E traffic can be load-balanced using short-path forwarding on the backup devices.

Before configuring VRRP-E load-balancing, VRRP-E must be configured on all devices in the VRRP-E session.

Perform this task on all backup VRRP-E Layer 3 devices to allow load sharing within a VRRP extended group.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an associated RBridge ID to enter RBridge configuration mode for a specific ID.

   ```
   device(config)# rbridge-id 122
   ```

3. Globally enable VRRP-E.

   ```
   device(config-rbridge-id-122)# protocol vrrp-extended
   device(config)# protocol vrrp-extended
   ```

4. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config-rbridge-id-122)# interface ve 2019
   device(config)# interface ve 2019
   ```

   In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned with a VLAN number of 2019.

5. Enter an IP address for the interface using the **ip address** command.

   ```
   device(config-ve-2019)# ip address 192.168.4.1/24
   ```

6. Enter the **vrrp-extended-group** command with a number to assign a VRRP-E group to the device.

   ```
   device(config-ve-2018)# vrrp-extended-group 19
   ```

   In this example, VRRP-E group configuration mode is entered.

7. Enter the **short-path-forwarding** command with a **revert-priority** value to configure the backup VRRP-E as an alternate path with a specified priority.

   ```
   device(config-vrrp-extended-group-19)# short-path-forwarding revert-priority 50
   ```

   When the backup device priority is higher than the configured **revert-priority** value, the backup router is able to perform short-path forwarding. If the backup priority is lower than the revert priority, short-path forwarding is disabled.

In the following example, short-path forwarding is configured on a backup VRRP-E device and a revert priority threshold is configured. If the backup device priority falls below this threshold, short-path forwarding is disabled.

```
device# configure
device(config)# rbridge-id 122
device(config-rbridge-id-122)# protocol vrrp-extended
device(config-rbridge-id-122)# interface ve 2019
device(config-ve-2019)# ip address 192.168.4.1/24
device(config-ve-2019)# vrrp-extended-group 19
device(config-vrrp-extended-group-19)# short-path-forwarding revert-priority 50

device# configure
device(config)# protocol vrrp-extended
device(config)# interface ve 2019
device(config-ve-2019)# ip address 192.168.4.1/24
device(config-ve-2019)# vrrp-extended-group 19
device(config-vrrp-extended-group-19)# short-path-forwarding revert-priority 50
```

Configuring VRRP-E load-balancing using short-path forwarding

VRRP-E traffic can be load-balanced using short-path forwarding on the backup devices.

Before configuring VRRP-E load-balancing, VRRP-E must be configured on all devices in the VRRP-E session.

Perform this task on all backup VRRP-E Layer 3 devices to allow load sharing within a VRRP extended group.

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

   ```
   device# configure terminal
   ```

2. To globally enable VRRP-E, enter the **router vrrp-extended** command.

   ```
   device(config)# router vrrp-extended
   ```

3. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config-vrrpe-router)# interface ve 10
   ```

   In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned a VLAN number of 10.

4. Enter an IP address for the interface using the **ip address** command.

   ```
   device(config-vif-10)# ip address 192.168.4.1/24
   ```

5. Enter the **ip vrrp-extended vrid** command with a number to assign a VRRP-E virtual router ID to the device.

   ```
   device(config-vif-10)# ip vrrp-extended vrid 5
   ```

   In this example, VRRP-E group configuration mode is entered.

6. Enter the **backup** command with a **priority** value to configure the device as a VRRP-E backup device.

   ```
   device(config-vif-10-vrid-5)# backup priority 50
   ```

7. Enter the **ip-address** command with an IP address that is not used on any VRRP-E device interface to add a virtual IP address to the VRRP-E instance.

   ```
   device(config-vif-10-vrid-5)# ip-address 192.168.4.254
   ```

8. Enter the **short-path-forwarding** command with a **revert-priority** value to configure the backup VRRP-E device as an alternate path with a specified priority.

   ```
   device(config-vif-10-vrid-5)# short-path-forwarding revert-priority 50
   ```

   When the backup device priority is higher than the configured **revert-priority** value, the backup router is able to perform short-path forwarding. If the backup priority is lower than the revert priority, short-path forwarding is disabled.

9. Enter the **activate** command to activate the VRRP-E instance.

   ```
   device(config-vif-10-vrid-5)# activate
   ```

In the following example, short-path forwarding is configured on a backup VRRP-E device, and a revert priority threshold is configured. If the backup device priority falls below this threshold, short-path forwarding is disabled.

```
device# configure terminal
device(config)# router vrrp-extended
device(config-vrrpe-router)# interface ve 10
device(config-vif-10)# ip address 192.168.4.1/24
device(config-vif-10)# ip vrrp-extended vrid 5
device(config-vif-10-vrid-5)# backup priority 50
device(config-vif-10-vrid-5)# ip-address 192.168.4.254
device(config-vif-10-vrid-5)# short-path-forwarding revert-priority 50
device(config-vif-10-vrid-5)# activate
```

VRRP-E slow start timer

In a VRRP extended (VRRP-E) configuration, if a master device goes offline, the backup router with the highest priority takes over after the expiration of the dead interval timer. When the original master device is back online, you can configure a slow-start timer interval that extends the time interval beyond the dead interval before the original master device transitions back to the role of master device.

The slow-start interval allows additional time for routing protocols, for example OSPF, to converge without causing route flapping during the transition from backup device to master device. Included in the VRRP-E slow-start timer feature are track port state changes and restart options. The **use-track-port** option implements a slow-start timer for the first tracked port "up" state change, in addition to the VRRP-E initialization state. The **restart** option restarts the slow-start timer for subsequent tracked port "up" state changes.

> NOTE
> If you change the backup priority of a VRRP-E backup router to be higher than the priority of the original master device, the slow-start timer will not work. The original master device will take over from the backup device immediately.

Configuring a VRRP-E slow-start timer

The slow-start timer is a VRRP-E interval timer that extends beyond the dead interval during a transition from the backup device that assumed the master role to the original master device that is back online and has a higher priority.

In a VRRP extended (VRRP-E) configuration, if a master device goes offline, the backup router with the highest priority takes over after the expiration of the dead interval timer. When the original master device is back online, you can configure a slow-start timer interval that extends the time interval beyond the dead interval before the original master device transitions back to the role of master device.

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

   ```
   device# configure terminal
   ```

2. To globally enable VRRP-E, enter the **router vrrp-extended** command.

   ```
   device(config)# router vrrp-extended
   ```

3. Enter the **slow-start** command with options to configure the interval, in seconds, and whether tracked-port state changes trigger the slow-start interval.

   ```
   device(config-vrrpe-router)# slow-start 40 use-track-port restart
   ```

In this example, the slow-start timer interval is set to 40 seconds, and the slow-start timer also runs after the first and subsequent tracked-port state changes.

```
device# configure terminal
device(config)# router vrrp-extended
device(config-vrrpe-router)# slow-start 40 use-track-port restart
```

Multiple virtual IP address support for VRRP-E

Support for multiple virtual IPv4 addresses per interface is available for a VRRP extended (VRRP-E) router instance. For an interface that has multiple subnet IPv4 addresses, a single VRRP-E provides redundancy when all the IP addresses are configured for the VRRP-E instance.

All virtual IP addresses must be configured before activating the VRRP-E instance. If a virtual IP address is to be added, the VRRP-E router instance must be disabled, and the configuration changes made before reactivating the VRRP-E instance. Between master and backup VRRP-E devices the virtual IP address configuration must be consistent to allow the generation of the same virtual MAC address for a specific virtual router ID (VRID). Gratuitous ARP messages are sent for all the configured IP addresses of each VRRP-E instance. To avoid regeneration of the virtual MAC address when a lower value virtual IPv4 address is configured, configure a static virtual MAC address for the VRRP-E router instance.

NOTE
Multiple virtual IPv4 address support is not compatible with the VRRP-E scaling feature.

Configuring multiple virtual IP addresses for VRRP-E

Configuring multiple virtual IPv4 addresses for an interface assigned as a VRRP extended (VRRP-E) router instance.

For an interface that has multiple subnet IPv4 addresses, a single VRRP-E provides redundancy when all the IP addresses are configured for the VRRP-E instance. In this task, configuring a static MAC address is an optional step.

1. From privileged EXEC mode, enter configuration mode by issuing the **configure terminal** command.

   ```
   device# configure terminal
   ```

2. Globally enable the VRRP-E protocol by entering the **router vrrp-extended** command.

   ```
   device(config)# router vrrp-extended
   ```

3. Configure the ethernet interface for the VRRP-E instance.

   ```
   device(config)# interface ethernet 1/1
   ```

4. Configure an IP address on the interface.

   ```
   device(config-if-e1000-1/1)# ip address 10.10.10.1/24
   ```

5. Repeat Step 4 for all IP addresses on the interface.

   ```
   device(config-if-e1000-1/1)# ip address 10.20.20.1/24
   ```

6. Configure a VRRP extended instance using a virtual routing ID (VRID).

   ```
   device(config-if-e1000-1/1)# ip vrrp-extended vrid 2
   ```

7. Configure the device as a backup VRRP-E device for VRID 2.

   ```
   device(config-if-e1000-1/1-vrid-2)# backup
   ```

8. Configure an optional static MAC address for VRID 2.

   ```
   device(config-if-e1000-1/1-vrid-2)# virtual-mac aaaa.bbbb.cccc
   ```

9. Configure a virtual IP address for VRID 2.

```
device(config-if-e1000-1/1-vrid-2)# ip-address 10.10.10.10
```

10. Repeat Step 9 to configure all the virtual IP addresses for VRID 2.

```
device(config-if-e1000-1/1-vrid-2)# ip-address 10.20.20.20
```

If you want to add a new virtual IP address after the initial setup, you must first deactivate the virtual router and then reconfigure all the virtual IP addresses.

11. Activate the virtual IP address or addresses for VRID 2.

```
device(config-if-e1000-1/1-vrid-2)# activate
```

The following example configures seven virtual IP addresses for VRRP-E virtual router instance 2.

```
device# configure terminal
device(config)# router vrrp-extended
device(config)# interface ethernet 1/1
device(config-if-e1000-1/1)# ip address 10.10.10.1/24
device(config-if-e1000-1/1)# ip address 10.20.20.1/24
device(config-if-e1000-1/1)# ip address 10.30.30.1/24
device(config-if-e1000-1/1)# ip address 10.40.40.1/24
device(config-if-e1000-1/1)# ip address 10.50.50.1/24
device(config-if-e1000-1/1)# ip address 10.60.60.1/24
device(config-if-e1000-1/1)# ip address 10.70.70.1/24
device(config-if-e1000-1/1)# ip vrrp-extended vrid 2
device(config-if-e1000-1/1-vrid-2)# backup
device(config-if-e1000-1/1-vrid-2)# virtual-mac aaaa.bbbb.cccc
device(config-if-e1000-1/1-vrid-2)# ip-address 10.10.10.10
device(config-if-e1000-1/1-vrid-2)# ip-address 10.20.20.20
device(config-if-e1000-1/1-vrid-2)# ip-address 10.30.30.30
device(config-if-e1000-1/1-vrid-2)# ip-address 10.40.40.40
device(config-if-e1000-1/1-vrid-2)# ip-address 10.50.50.50
device(config-if-e1000-1/1-vrid-2)# ip-address 10.60.60.60
device(config-if-e1000-1/1-vrid-2)# ip-address 10.70.70.70
device(config-if-e1000-1/1-vrid-2)# activate
```

Displaying multiple virtual IP addresses for VRRP-E information

Displays information about the multiple virtual IPv4 addresses that are configured for a VRRP-E instance.

Several options of the **show ip vrrp-extended** command can display information about the multiple virtual IP addresses configured for a VRRP-E instance. Use the steps below in any order.

1. Enter the **show ip vrrp-extended brief** command to display the number of configured virtual IPv4 addresses for each VRRP-E router instance and the virtual IPv4 addresses.

```
device# show ip vrrp-extended brief

Total number of VRRP-Extended routers defined: 3
Flags Codes - P:Preempt 2:V2 3:V3
Short-Path-Fwd Codes - ER: Enabled with revertible option, RT: reverted,
                       NR: not reverted

Intf VRID Curr Flags State  MasterIP BackupIP (No) VirtualIP Short-   Track MCT
          Prio              Address  Address       Address   Path-Fwd VPLS-State
-------------------------------------------------------------------------------
1/1  1    100  P2    Master Local    Unknown  ( 7) 10.10.10.10 Enabled Disable
                                                   10.20.20.20
                                                   10.30.30.30
                                                   10.40.40.40
                                                   10.50.50.50
                                                   10.60.60.60
                                                   10.70.70.70
```

2. Enter the **show ip vrrp-extended** command with a specific VRID and interface to display the number of configured IPv4 addresses for each VRRP-E router instance and the actual IPv4 addresses in a different format with the detailed information about each virtual router instance.

```
device# show ip vrrp-extended vrid 1 ethernet 1/1

Interface 1/1
----------------
 auth-type no authentication
 VRID 1 (index 1)
 interface 1/1
 state master
 administrative-status enabled
 mode non-owner(backup)
 virtual mac 02e0.527d.7c01
 priority 100
 current priority 100
 track-priority 5
 hello-interval 1 sec
 backup hello-interval 60 sec
 slow-start timer (configured) 0 sec
 advertise backup disabled
 dead-interval 3500 ms
 preempt-mode true
 number of configured virtual address 7
 virtual ip address 10.10.10.10 10.20.20.20 10.30.30.30 10.40.40.40
                    10.50.50.50 10.60.60.60 10.70.70.70
 next hello sent in 1000 ms
 Track MCT-VPLS-State: Disable
 short-path-forwarding enabled
```

VRRP-E scaling using logical groups

Scaling the number of VRRP extended (VRRP-E) instances is allowed using a grouping mechanism. VRRP-E instances are configured into logical groups consistently across all the VRRP-E master and backup devices.

> **NOTE**
> The XMR and MLX Series platforms support scaling up to 4000 VRRP-E instances.
> The CER 2000 Series platform supports scaling up to 255 VRRP-instances

Each VRRP-E logical group is assigned with a group master and the configuration is performed to identify the group master for each VRRP-E virtual router ID (VRID) configured on a device. Group members stop advertising hello messages and they inherit the state of the group master. The significant reduction in load on the CPU when the hello advertisements are not sent or processed, allows more VRRP-E instances to be configured.

A gratuitous ARP router advertisement is sent from the group-master on behalf of its group members once every 30 seconds by default to allow the devices on the network to learn the virtual MAC address of the group master. The gratuitous ARP router advertisement interval can be configured.

> **NOTE**
> If the maximum number of VRRP-E instances exist, configuring a gratuitous ARP router advertisement interval lower than 5 seconds can result in high CPU usage.

The configuration for the following commands is inherited from the group master device configuration. VRRP-E ignores the configuration of any parameter that is to be inherited from the group master.

- advertise
- auth-type
- backup
- backup-hello-interval
- dead-interval
- hello-interval
- non-preempt-mode
- use-v2-checksum

The configuration for the following commands is not inherited from the group master. Group members retain their individual configuration parameters from these commands.

- activate
- enable
- ip-address
- ipv6-address
- short-path-forwarding
- virtual-mac

> **NOTE**
> The VRRP-E scaling feature is not compatible with the VRRP-E multiple virtual IP addresses feature.

Configuring VRRP-E scaling

VRRP-E scaling is implemented by configuring VRRP-E instances into logical groups using a group master and by configuring a gratuitous ARP router advertisement interval.

This task has to be repeated for all VRRP-E instances that are to be part of the same group. VRRP-E instances are always physically grouped on the same device and the logical group is maintained consistently across all devices. VRRP-E scaling is supported for IPv4 and IPv6 VRRP-E sessions.

NOTE
The VRRP-E scaling feature is not compatible with the VRRP-E multiple virtual IP addresses feature.

1. From privileged EXEC mode, enter configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable the IPv6 VRRP-E protocol by entering the **ipv6 router vrrp-extended** command.

```
device(config)# ipv6 router vrrp-extended
```

3. Configure the gratuitous ARP interval for the IPv6 VRRP-E instance.

```
device(config-ipv6-VRRP-E-router)# garp-ra-interval 90
```

NOTE
In IPv4 VRRP-E the router prompt does not display the "-VRRP-E-router" text.

4. Configure the ethernet interface for the IPv6 VRRP-E instance.

```
device(config-ipv6-VRRP-E-router)# interface ethernet 1/5
```

5. Configure an IPv6 address on the interface.

```
device(config-if-e1000-1/5)# ipv6 address 3013::2/64
```

6. Configure an IPv6 VRRP extended instance using a virtual routing ID (VRID).

```
device(config-if-e1000-1/5)# ipv6 vrrp-extended vrid 2
```

7. Configure the device as a backup VRRP-E device for VRID 2.

```
device(config-if-e1000-1/5-ipv6-vrid-2)# backup priority 50 track-priority 20
```

8. Configure a virtual link-local IPv6 address for VRID 2.

```
device(config-if-e1000-1/5-ipv6-vrid-2)# ipv6-address fe80::768e:f8ff:fe2a:0099
```

9. Configure a virtual IPv6 address for VRID 2.

```
device(config-if-e1000-1/5-ipv6-vrid-2)# ipv6-address 3013::99
```

10. Configures the VRRP-E master for a group for VRID 2.

```
device(config-if-e1000-1/5-ipv6-vrid-2)# group-master interface ethernet 1/2 vrid 1
```

In this example,

11. Activate the virtual IP address or addresses for VRID 2.

```
device(config-if-e1000-1/5-ipv6-vrid-2)# activate
```

The following example configures virtual router 1 on interface Ethernet 1/2 as the VRRP-E group master of virtual router 2 on interface Ethernet 1/5. Virtual router 2 will stop sending and receiving VRRP-E packets because VRID 1 now maintains the session for virtual router 2 by sending and receiving VRRP-E packets. The interval between gratuitous ARP messages is set to 90 seconds.

```
device# configure terminal
device(config)# router ipv6 vrrp-extended
device(config-ipv6-VRRP-E-router)# garp-ra-interval 90
device(config-ipv6-VRRP-E-router)# interface ethernet 1/5
device(config-if-e1000-1/5)# ipv6 address 3013::2/64
device(config-if-e1000-1/5)# ipv6 vrrp-extended vrid 2
device(config-if-e1000-1/5-ipv6-vrid-2)# backup priority 50 track-priority 20
device(config-if-e1000-1/5-ipv6-vrid-2)# ipv6-address fe80::768e:f8ff:fe2a:0099
device(config-if-e1000-1/5-ipv6-vrid-2)# ipv6-address 3013::99
device(config-if-e1000-1/5-ipv6-vrid-2)# group-master interface ethernet 1/2 vrid 1
device(config-if-e1000-1/5-ipv6-vrid-2)# activate
```

Displaying VRRP-E scaling information

Displays information about VRRP-E group members and group masters configured for the VRRP-E scaling feature.

Several options of the **show ip vrrp-extended** or **show ipv6 vrrp-extended** commands can display information about the VRRP-E scaling feature configuration. Use the steps below in any order.

1. Enter the **show ip vrrp-extended** or the **show ipv6 vrrp-extended** command with the **vrid** option to display group member information for the VRRP-E scaling feature for a specific VRID. In the example below, output for VRID 1 shows the total number of group members and in which VRIDs the members are configured.

```
device(config)# show ip vrrp-extended vrid 1

VRID 1 (index 1)
 interface 1/1
 state master
. administrative-status enabled
.
.
 group-member count 3
 group-members
   ethernet 1/2 vrid 2
   ethernet 1/2 vrid 3
   ethernet 1/2 vrid 4
```

2. Enter the **show ipv6 vrrp-extended** or the **show ip vrrp-extended** command with a specific VRID and interface to display group master information for the VRRP-E scaling feature for a specific interface. Only partial output is displayed.

```
device# show ipv6 vrrp-extended ve 100 vrid 2

VRID 2 (index 2)
 interface v100
 state backup
.
.
.
 group-master ve 100 vrid 1
```

Displaying VRRPv2 information

Various show commands can be used to display statistical and summary information about VRRP and VRRP-E configurations.

Before displaying VRRP information, VRRPv2 must be configured and enabled in your VRRP or VRRP-E network to generate traffic.

Use one or more of the following commands to display VRRPv2 information. The commands do not have to be entered in this order.

1. Enter the **show vrrp** command with a virtual-group ID to display detailed information about one virtual group ID.

```
device# show vrrp 1

============Rbridge-id:2============

Total number of VRRP session(s)   : 1

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
  Authentication type: No Authentication
  State: Initialize
  Session Master IP Address:
  Virtual IP(s): 192.168.4.1
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: ENABLE  (default: ENABLE)
  Hold time: 0 sec  (default: 0 sec)
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========
  Statistics:
    Advertisements: Rx: 60, Tx: 6
    Gratuitous ARP: Tx: 2
```

   This example output shows that one IPv4 VRRP session is configured.

2. Enter the **show vrrp** command with a virtual-group ID to display detailed information about one virtual group ID.

```
device# show vrrp 1

Total number of VRRP session(s)   : 1

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
  Authentication type: No Authentication
  State: Initialize
  Session Master IP Address:
  Virtual IP(s): 192.168.4.1
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: ENABLE  (default: ENABLE)
  Hold time: 0 sec  (default: 0 sec)
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========
  Statistics:
    Advertisements: Rx: 60, Tx: 6
    Gratuitous ARP: Tx: 2
```

   This example output shows that one IPv4 VRRP session is configured.

3. Enter the **show ip vrrp** command with the **vrid** option and a virtual router ID (VRID) to display IPv4 VRRP configuration information about VRID 1.

```
device# show ip vrrp vrid 1

Interface 1/1
----------------
auth-type no authentication
VRID 1 (index 1)
interface 1/1
state master
administrative-status enabled
version v2
mode owner
virtual mac aaaa.bbbb.cccc (configured)
priority 255
current priority 255
track-priority 2
hello-interval 1 sec
backup hello-interval 6
```

4. Enter the **show vrrp summary** command.

```
device# show vrrp summary

============Rbridge-id:2============

Total number of VRRP session(s)   : 1
Master session count  : 1
Backup session count  : 0
Init session count    : 0

VRID  Session  Interface       Admin     Current   State    Short-path  Revert    SPF
                               State     Priority           Forwarding  Priority  Reverted
====  =======  =========       =====     ========  ======   ==========  ========  ========
1     VRRP     Ve 100          Enabled   110       Master
```

This example displays information about VRRP sessions.

5. Enter the **show vrrp summary** command.

```
device# show vrrp summary

Total number of VRRP session(s)   : 1
Master session count  : 1
Backup session count  : 0
Init session count    : 0

VRID  Session  Interface       Admin     Current   State    Short-path  Revert    SPF
                               State     Priority           Forwarding  Priority  Reverted
====  =======  =========       =====     ========  ======   ==========  ========  ========
1     VRRP     Ve 100          Enabled   110       Master
```

This example displays information about VRRP sessions.

6. Enter the **show ip vrrp brief** command.

```
device(config)# show ip vrrp brief

Total number of VRRP routers defined: 2
Flags Codes - P:Preempt 2:V2 3:V3 S:Short-Path-Fwd
Inte- VRID  Current  Flags    State   Master IP Backup IP  Virtual IP
rface       Priority                   Address   Address    Address
----------------------------------------------------------------------
1/1   10    255      P2-      Master  Local     Unknown    10.30.30.2
1/3   13    100      P2-      Master  Local     Unknown    10.13.13.3
```

This example displays summary information about VRRP sessions.

7. Enter the **show vrrp interface** command with interface ve 10 options and detailed output.

```
device# show vrrp int ve 10 detail

============Rbridge-id:2============

Total number of VRRP session(s)   : 1

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
  Authentication type: No Authentication
  State: Initialize
  Session Master IP Address:
  Virtual IP(s): 192.168.4.1
  Virtual MAC Address: 0000.5e00.0101
  Configured Priority: 110 (default: 100); Current Priority: unset
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: ENABLE  (default: ENABLE)
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 60, Tx: 6
    Gratuitous ARP             : Tx: 2
    Session becoming master    : 0
    Advts with wrong interval  : 0
    Prio Zero pkts             : Rx: 0, Tx: 0
    Invalid Pkts Rvcd          : 0
    Bad Virtual-IP Pkts        : 0
    Invalid Authenticaton type : 0
    Invalid TTL Value          : 0
    Invalid Packet Length      : 0
```

8. Enter the **show vrrp interface** command with interface ve 10 options and detailed output.

```
device# show vrrp int ve 10 detail

Total number of VRRP session(s)   : 1

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
  Authentication type: No Authentication
  State: Initialize
  Session Master IP Address:
  Virtual IP(s): 192.168.4.1
  Virtual MAC Address: 0000.5e00.0101
  Configured Priority: 110 (default: 100); Current Priority: unset
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: ENABLE  (default: ENABLE)
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 60, Tx: 6
    Gratuitous ARP             : Tx: 2
    Session becoming master    : 0
    Advts with wrong interval  : 0
    Prio Zero pkts             : Rx: 0, Tx: 0
    Invalid Pkts Rvcd          : 0
    Bad Virtual-IP Pkts        : 0
    Invalid Authenticaton type : 0
    Invalid TTL Value          : 0
    Invalid Packet Length      : 0
```

9. Enter the **show ip vrrp-extended statistics** command for Ethernet interface 1/5.

```
device# show ip vrrp-extended interface Ethernet 1/5

Interface 1/5
----------------
VRID 2
- number of transitions to backup state = 1
- number of transitions to master state = 1
- total number of vrrp-extended packets received = 0
  . received backup advertisements = 0
  . received packets with zero priority = 0
  . received packets with invalid type = 0
  . received packets with invalid authentication type = 0
  . received packets with authentication type mismatch = 0
  . received packets with authentication failures = 0
  . received packets dropped by owner = 0
  . received packets with ttl errors = 0
  . received packets with ipv6 address mismatch = 0
  . received packets with advertisement interval mismatch = 0
  . received packets with invalid length = 0
- total number of vrrp-extended packets sent = 2004
  . sent backup advertisements = 0
  . sent packets with zero priority = 0
- received neighbor solicitation packets dropped = 0
- received proxy neighbor solicitation packets dropped = 0
- received ip packets dropped = 0
```

Clearing VRRPv2 statistics

VRRPv2 session counters can be cleared using a CLI command.

Ensure that VRRPv2 or VRRP-Ev2 is configured and enabled in your network.

To determine the effect of clearing the VRRP statistics, an appropriate **show** command is entered before and after the **clear** command.

1. Enter the **end** or **exit** command to return to privileged EXEC mode.
2. Enter the **show vrrp** command with a virtual-group ID.

```
device# show vrrp 1

============Rbridge-id:125============

Total number of VRRP session(s)   : 2

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
.
.
.
Statistics:
    Advertisements: Rx: 0, Tx: 60
    Neighbor Advertisements: Tx: 30
```

3. Enter the **show vrrp** command with a virtual-group ID.

```
device# show vrrp 1

Total number of VRRP session(s)   : 2

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
.
.
.
Statistics:
    Advertisements: Rx: 0, Tx: 60
    Neighbor Advertisements: Tx: 30
```

4. Enter the **show ip vrrp statistics** command for Ethernet interface 1/5.

```
device# show ip vrrp statistics ethernet 1/5

Interface 1/5
----------------
VRID 2
- number of transitions to backup state = 1
- number of transitions to master state = 1
- total number of vrrp packets received = 0
  . received backup advertisements = 0
  . received packets with zero priority = 0
.
.
.
- total number of vrrp packets sent = 2004
  . sent backup advertisements = 6
  . sent packets with zero priority = 0
- received neighbor solicitation packets dropped = 0
```

5. Enter the **clear vrrp statistics** command.

```
device# clear vrrp statistics
```

6. Enter the **clear ip vrrp statistics** command.

```
device# clear ip vrrp statistics
```

7. Enter the **show vrrp** command with a virtual-group ID.

```
device# show vrrp 1

============Rbridge-id:125============

Total number of VRRP session(s)   : 2

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
.
.
.
Statistics:
    Advertisements: Rx: 0, Tx: 6
    Neighbor Advertisements: Tx: 3
```

In this show output after the **clear vrrp statistics** command has been entered, you can see that the statistical counters have been reset. Although some of the counters are showing numbers because VRRP traffic is still flowing, the numbers are much lower (6 transmissions instead of 60 transmissions) than in the initial **show vrrp** command output.

8. Enter the **show vrrp** command with a virtual-group ID.

```
device# show vrrp 1

Total number of VRRP session(s)   : 2

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
.
.
.
Statistics:
    Advertisements: Rx: 0, Tx: 6
    Neighbor Advertisements: Tx: 3
```

In this show output after the **clear vrrp statistics** command has been entered, you can see that the statistical counters have been reset. Although some of the counters are showing numbers because VRRP traffic is still flowing, the numbers are much lower (6 transmissions instead of 60 transmissions) than in the initial **show vrrp** command output.

9. Enter the **show ip vrrp statistics** command for Ethernet interface 1/5.

```
device# show ip vrrp statistics ethernet 1/5

Interface 1/5
----------------
VRID 2
- number of transitions to backup state = 0
- number of transitions to master state = 0
- total number of vrrp packets received = 0
  . received backup advertisements = 0
  . received packets with zero priority = 0
.
.
.
- total number of vrrp packets sent = 6
  . sent backup advertisements = 0
  . sent packets with zero priority = 0
- received neighbor solicitation packets dropped = 0
```

In this show output for a specified interface after the **clear ip vrrp statistics** command has been entered, you can see that the statistical counters have been reset. Although some of the counters are showing numbers because VRRP traffic is still flowing, the numbers are much lower (8 transmissions instead of 2004 transmissions) than in the initial **show ip vrrp statistics** command output.

VRRPv3

- VRRPv3 overview ... 691
- Enabling IPv6 VRRPv3 ... 692
- Enabling an IPv6 VRRPv3 owner device ... 694
- Enabling an IPv6 VRRPv3 backup device ... 695
- Enabling IPv4 VRRPv3 ... 696
- Enabling an IPv4 VRRPv3 owner device ... 698
- Enabling an IPv4 VRRPv3 backup device ... 699
- Tracked ports and track priority with VRRP and VRRP-E ... 700
- Tracked IPsec tunnels and track priority with VRRP and VRRP-E ... 703
- VRRP hold timer ... 706
- Accept mode for backup VRRP devices ... 708
- Accept mode for backup VRRP devices ... 709
- Accept mode for backup VRRP devices ... 711
- Alternate VRRPv2 checksum for VRRPv3 IPv4 sessions ... 711
- VRRPv3 router advertisement suppression ... 714
- Automatic generation of a virtual link-local address for VRRPv3 ... 716
- Displaying VRRPv3 statistics ... 718
- Clearing VRRPv3 statistics ... 721
- VRRP-Ev3 Overview ... 724
- Enabling IPv6 VRRP-Ev3 ... 724
- Enabling an IPv6 VRRP-Ev3 device ... 725
- Configuring MD5 authentication on IPv6 VRRP-Ev3 interfaces ... 727
- VRRP-E load-balancing using short-path forwarding ... 728
- VRRP-Ev3 sub-second failover ... 731
- Displaying and clearing VRRP-Ev3 statistics ... 732

VRRPv3 overview

VRRP version 3 (VRRPv3) introduces IPv6 address support for both standard VRRP and VRRP enhanced (VRRP-E).

Virtual Router Redundancy Protocol (VRRP) is designed to eliminate the single point of failure inherent in a static default routed environment by providing redundancy to Layer 3 devices within a local area network (LAN). VRRP uses an election protocol to dynamically assign the default gateway for a host to one of a group of VRRP routers on a LAN. Alternate gateway router paths can be allocated without changing the IP address or MAC address by which the host device knows its gateway.

VRRPv3 implements support for IPv6 addresses for networks using IPv6, and it also supports IPv4 addresses for dual-stack networks configured with VRRP or VRRP-E. VRRPv3 is compliant with RFC 5798. The benefit of implementing VRRPv3 is faster switchover to backup devices than can be achieved using standard IPv6 neighbor discovery mechanisms. With VRRPv3, a backup router can become a master router in a few seconds with less overhead traffic and no interaction with the hosts.

When VRRPv3 is configured, the master device that owns the virtual IP address and a master device that does not own the virtual IP address can both respond to ICMP echo requests (using the **ping** command) and accept Telnet and other management traffic sent to the virtual IP address. In VRRPv2, only a master device on which the virtual IP address is the address of an interface on the master device can respond to ping and other management traffic.

The following are other IPv6 VRRPv3 functionality details:

- VRRPv2 functionality is supported by VRRPv3 except for VRRP authentication.

- Two VRRP and VRRP-E sessions cannot share the same group ID on the same interface.

  NOTE
  When implementing IPv6 VRRPv3 across a network with devices from other vendors, be aware of a potential interoperability issue with IPv6 VRRPv3 and other vendor equipment. Extreme has implemented IPv6 VRRPv3 functionality to comply with RFC 5798 and will interoperate comfortably with other vendors that support RFC 5798.

VRRPv3 functionality differences on VDX devices

The implementation of VRRPv3 varies across the VDX products.

VRRPv3 functionality on the VDX 8770 platforms.

- VRRP and VRRP-E configurations can coexist on the same interface.

VRRPv3 functionality on the VDX 6740 platforms.

- VRRP and VRRP-E configurations cannot coexist on the same interface.

VRRPv3 functionality on the VDX 6940 platforms.

- IPv4 VRRP or IPv6 VRRP can be individually configured along with VRRP-E IPv4 and IPv6, but not IPv4 VRRP and IPv6 VRRP simultaneously.
- If IPv4 VRRP and IPv6 VRRP are both configured, then VRRP-E cannot be configured.

VRRPv3 performance and scalability metrics for Network OS devices

The following table defines VRRPv3 system resource metrics by Network OS device.

TABLE 48 System resource metrics for VRRPv3

System resource	VDX 87xx	VDX 67xx	VDX 6940-36Q, 6940-144S
Max # of VRRP and VRRP-E sessions with advertisement interval of 1 second	1024	255	512
Max # of VRRP and VRRP-E sessions with advertisement interval of 500 milliseconds	500	100	200
Max # of VRRP and VRRP-E sessions with advertisement interval of 200 milliseconds	200	50	100
VRRP sessions per interface	32	32	16
Max # of VRRP devices	8	8	8

Enabling IPv6 VRRPv3

IPv6 VRRPv3 is enabled on a device when a virtual IPv6 address is assigned to a VRRPv3 group.

Before assigning a virtual IPv6 address to a VRRPv3 group, you must configure IPv6 VRRP version 3 on a virtual Ethernet interface and assign a VRRPv3 group to the device. The VRRPv3 session is enabled using a virtual IPv6 address. The device must be a router or another device that supports Layer 3 routing.

Perform this task on all devices that are to run VRRPv3. The device to which the virtual IP address belongs determines the initial master device status with all the other devices acting as backups.

1. Enter the **configure** command to access global configuration mode.

   ```
   device# configure
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 125
   ```

3. To globally enable VRRPv3, enter the **ipv6 protocol vrrp** command.

   ```
   device(config-rbridge-id-125)# ipv6 protocol vrrp
   device(config)# ipv6 protocol vrrp
   ```

4. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config-rbridge-id-125)# interface ve 2018
   device(config)# interface ve 2018
   ```

   In this example, virtual Ethernet (ve) interface configuration mode is entered and the interface is assigned with a VLAN number of 2018.

5. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2018)# ipv6 address 2001:2018:8192::125/64
   ```

6. Enter the **ipv6 vrrp-group** command with a number to assign a VRRPv3 group to the device.

   ```
   device(config-ve-2018)# ipv6 vrrp-group 18
   ```

   In this example, VRRP group configuration mode is entered.

7. Enter the **virtual-ip** command to assign a link-local virtual IPv6 address to a VRRPv3 group.

   ```
   device(config-vrrp-group-18)# virtual-ip fe80::2018:1
   ```

   In this example, the link-local IPv6 address of the virtual router is assigned to VRRPv3 group 18. The first virtual IP address entered enables the VRRPv3 session.

   > NOTE
   > A link-local IPv6 address is valid only for a single network link. If the virtual IP address can be reached from outside the local network, a global IPv6 address must be configured as a virtual IP address. At least one link-local address is also required.

8. Enter the **virtual-ip** command to assign a virtual IPv6 address to a VRRPv3 group.

   ```
   device(config-vrrp-group-18)# virtual-ip 2001:2018:8192::1
   ```

   In this example, the IPv6 address of the virtual router is assigned to VRRPv3 group 18.

The following example shows how to enable a VRRPv3 session by assigning virtual IP addresses to a VRRPv3 virtual group.

```
device# configure
device(config)# rbridge-id 125
device(config-rbridge-id-125)# ipv6 protocol vrrp
device(config-rbridge-id-125)# interface ve 2018
device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
device(config-ve-2018)# ipv6 vrrp-group 18
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
device(config-vrrp-group-18)# virtual-ip 2001:2018:8192::1

device# configure
device(config)# ipv6 protocol vrrp
device(config)# interface ve 2018
device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
device(config-ve-2018)# ipv6 vrrp-group 18
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
device(config-vrrp-group-18)# virtual-ip 2001:2018:8192::1
```

Enabling an IPv6 VRRPv3 owner device

This task is performed on the device that is designated as the owner VRRP device because the IPv6 address of one of its physical interfaces is assigned as the IP address of the virtual router. For each VRRP session, there are master and backup routers, and the owner router is elected, by default, as the master router.

NOTE
When implementing IPv6 VRRPv3 across a network with devices from other vendors, be aware of a potential interoperability issue. IPv6 VRRPv3 functionality is implemented to comply with RFC 5798 and will interoperate well with other vendors that support RFC 5798.

1. On the device designated as the owner VRRP device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

   ```
   device# configure terminal
   ```

2. Before enabling IPv6 VRRP, you must globally enable IPv6 routing.

   ```
   device(config)# ipv6 unicast-routing
   ```

3. Globally enable IPv6 VRRP.

   ```
   device(config)# ipv6 router vrrp
   ```

4. Configure the Ethernet interface link for the owner device.

   ```
   device(config)# interface ethernet 1/5
   ```

5. Configure the IPv6 address of the interface.

   ```
   device(config-if-e1000-1/5)# ipv6 address fd2b::2/64
   ```

6. Assign the owner device to the virtual router ID (VRID) 2.

   ```
   device(config-if-e1000-1/5)# ipv6 vrrp vrid 2
   ```

   NOTE
   You can assign a VRID number in the range of 1 through 255.

7. Designate this router as the VRRP owner device.

```
device(config-if-e1000-1/5-vrid-2)# owner
```

8. Assign an IPv6 link-local address to the VRID for use in the local network.

```
device(config-if-e1000-1/5-vrid-2)# ipv6-address fe80::768e:f8ff:fe2a:0099
```

9. Assign a global IPv6 address to the VRID.

```
device(config-if-e1000-1/5-vrid-2)# ipv6-address fd2b::2
```

10. Enable the VRRP session.

```
device(config-if-e1000-1/5-vrid-2)# activate
```

The following example configures a VRRP owner device.

```
device# configure terminal
device(config)# ipv6 unicast-routing
device(config)# ipv6 router vrrp
device(config-ipv6-vrrp-router)# interface ethernet 1/5
device(config-if-e1000-1/5)# ipv6 address fd2b::2/64
device(config-if-e1000-1/5)# ipv6 vrrp vrid 2
device(config-if-e1000-1/5-vrid-2)# owner
device(config-if-e1000-1/5-vrid-2)# ipv6-address fe80::768e:f8ff:fe2a:0099
device(config-if-e1000-1/5-vrid-2)# ipv6-address fd2b::2
device(config-if-e1000-1/5-vrid-2)# activate
```

Enabling an IPv6 VRRPv3 backup device

This task is performed on all devices that are designated as backup VRRPv3 devices. Initially a backup priority is set to 100. For each VRRPv3 session, there are master and backup routers, and the IPv6 address assigned here to the VRID is the IPv6 address of the master router. The task is repeated on each backup VRRPv3 device with corresponding changes to the interface number and IPv6 address of the interface.

NOTE
When implementing IPv6 VRRPv3 across a network with devices from other vendors, be aware of a potential interoperability issue. IPv6 VRRPv3 functionality is implemented to comply with RFC 5798 and will interoperate well with other vendors that support RFC 5798.

1. On the device designated as a backup VRRPv3 device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable IPv6 VRRP.

```
device(config)# ipv6 router vrrp
```

3. Configure the Ethernet interface link for the owner device.

```
device(config-ipv6-vrrp-router)# interface ethernet 1/4
```

4. Configure the IPv6 address of the interface.

```
device(config-if-e1000-1/4)# ipv6 address fd2b::3/64
```

5. Assign the backup device to the virtual router ID (VRID) 2.

```
device(config-if-e1000-1/4)# ipv6 vrrp vrid 2
```

NOTE
You can assign a VRID number in the range of 1 through 255.

6. Designate this router as a VRRPv3 backup device and assign it a priority of 100.

```
device(config-if-e1000-1/4-vrid-2)# backup priority 100
```

7. By default, backup VRRP devices do not send hello messages to advertise themselves to the master. Use the following command to enable a backup router to send hello messages to the master VRRP device.

```
device(config-if-e1000-1/4-vrid-2)# advertise backup
```

8. Assign the IPv6 link-local address to the VRID for use in the local network.

```
device(config-if-e1000-1/4-vrid-2)# ipv6-address fe80::768e:f8ff:fe2a:0099
```

9. Assign the global IPv6 address to the VRID.

```
device(config-if-e1000-1/4-vrid-2)# ipv6-address fd2b::2
```

10. Enable the VRRP session.

```
device(config-if-e1000-1/4-vrid-2)# activate
```

The following example configures an IPv6 VRRPv3 backup device.

```
device# configure terminal
device(config)# ipv6 router vrrp
device(config-ipv6-vrrp-router)# interface ethernet 1/4
device(config-if-e1000-1/4)# ipv6 address fd2b::3/64
device(config-if-e1000-1/4)# ipv6 vrrp vrid 2
device(config-if-e1000-1/4-vrid-2)# backup priority 100
device(config-if-e1000-1/4-vrid-2)# advertise backup
device(config-if-e1000-1/4-vrid-2)# ipv6-address fe80::768e:f8ff:fe2a:0099
device(config-if-e1000-1/4-vrid-2)# ipv6-address fd2b::2
device(config-if-e1000-1/4-vrid-2)# activate
```

Enabling IPv4 VRRPv3

IPv4 VRRPv3 is enabled on a device when a virtual IP address is assigned to a VRRPv3 group.

VRRPv3 supports IPv4 sessions as well as IPv6 sessions. To configure a VRRPv3 session for IPv4 assign a virtual router group with the **v3** option to the device. The device must be a router or another device that supports Layer 3 routing.

Perform this task on all devices that are to run IPv4 VRRPv3. The device to which the virtual IP address belongs determines the initial master device status with all the other devices acting as backups.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 101
```

3. To globally enable VRRP, enter the **protocol vrrp** command.

```
device(config-rbridge-id-101)# protocol vrrp
device(config)# protocol vrrp
```

4. Configure the Ethernet interface.

```
device(config-rbridge-id-101)# interface tengigabitethernet 101/1/6
device(config)# interface ethernet 1/6
device(config)# interface ethernet 0/6
```

   In this example, tengigabitethernet (te) interface configuration mode is entered and the interface is assigned with an associated RBridge ID and slot/port number of 101/1/6.

   In this example, Ethernet interface configuration mode is entered and the interface is assigned with a slot/port number of 1/6.

   In this example, Ethernet interface configuration mode is entered and the interface is assigned with a slot/port number of 0/6.

5. Enter an IPv4 address for the interface using the **ip address** command.

```
device(config-if-te-101/1/6)# ip address 192.168.5.2/24
device(conf-if-eth-1/6)# ip address 192.168.5.2/24
device(conf-if-eth-0/6)# ip address 192.168.5.2/24
```

6. Enter the **vrrp-group** command with a number to assign a virtual router group to the device and a version to configure VRRPv3.

```
device(config-if-te-101/1/6)# vrrp-group 10 version 3
device(conf-if-eth-1/6)# vrrp-group 10 version 3
device(conf-if-eth-0/6)# vrrp-group 10 version 3
```

   In this example, a VRRPv3 group is assigned and VRRP group configuration mode is entered.

7. Enter the **advertisement-interval** command with a number in milliseconds to configure the interval at which the master VRRP router advertises its existence to the backup routers.

```
device(config-vrrp-group-10)# advertisement-interval 2000
```

   In this example, the interval is expressed as 2000 milliseconds because VRRPv3 uses milliseconds instead of seconds for the advertisement interval.

8. Enter the **virtual-ip** command to assign a virtual IP address to a VRRPv3 group.

```
device(config-vrrp-group-10)# virtual-ip 192.168.5.2
```

   In this example, the IPv4 address of the virtual router is assigned to VRRPv3 group 10. This virtual IP address belongs to this device and this device will assume the role of the master device.

The following example shows how to enable an IPv4 VRRPv3 session by assigning virtual IP addresses to a VRRPv3 virtual group.

```
device# configure
device(config)# rbridge-id 101
device(config-rbridge-id-101)# protocol vrrp
device(config-rbridge-id-101)# interface tengigabitethernet 101/1/6
device(config-if-te-101/1/6)# ip address 192.168.5.2/24
device(config-if-te-101/1/6)# vrrp-group 10 version 3
device(config-vrrp-group-10)# advertisement-interval 2000
device(config-vrrp-group-10)# virtual-ip 192.168.5.2

device# configure
device(config)# protocol vrrp
device(config)# interface ethernet 1/6
device(config-if-eth-1/6)# ip address 192.168.5.2/24
device(config-if-eth-1/6)# vrrp-group 10 version 3
device(config-vrrp-group-10)# advertisement-interval 2000
device(config-vrrp-group-10)# virtual-ip 192.168.5.2

device# configure
device(config)# protocol vrrp
device(config)# interface ethernet 0/6
device(conf-if-eth-0/6)# ip address 192.168.5.2/24
device(conf-if-eth-0/6)# vrrp-group 10 version 3
device(config-vrrp-group-10)# advertisement-interval 2000
device(config-vrrp-group-10)# virtual-ip 192.168.5.2
```

Enabling an IPv4 VRRPv3 owner device

VRRPv3 supports IPv4 sessions as well as IPv6 sessions. To configure a VRRPv3 session for IPv4, assign a virtual router group with the VRRP version set to 3. This task is performed on the device that is designated as the owner VRRP device because the IP address of one of its physical interfaces is assigned as the IP address of the virtual router.

1. On the device designated as the owner VRRP device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

   ```
   device# configure terminal
   ```

2. Globally enable VRRP.

   ```
   device(config)# router vrrp
   ```

3. Configure an Ethernet interface.

   ```
   device(config)# interface ethernet 1/6
   ```

4. Configure the IP address of the interface.

   ```
   device(config-if-e1000-1/6)# ip address 10.53.5.1/24
   ```

5. Assign the virtual router ID (VRID) 1 to the interface.

   ```
   device(config-if-e1000-1/6)# ip vrrp vrid 1
   ```

   NOTE
   You can assign a VRID number in the range of 1 through 255.

6. Designate this router as the VRRP owner device.

   ```
   device(config-if-e1000-1/6-vrid-1)# owner
   ```

7. Configure the IP address of the VRID.

```
device(config-if-e1000-1/6-vrid-1)# ip-address 10.53.5.1
```

8. Enable the VRRP session.

```
device(config-if-e1000-1/6-vrid-1)# activate
```

The following example configures an IPv4 VRRPv3 owner device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/6
device(config-if-e1000-1/6)# ip address 10.53.5.1/24
device(config-if-e1000-1/6)# ip vrrp vrid 1
device(config-if-e1000-1/6-vrid-1)# owner
device(config-if-e1000-1/6-vrid-1)# ip-address 10.53.5.1
device(config-if-e1000-1/6-vrid-1)# activate
VRRP router 1 for this interface is activating
```

Enabling an IPv4 VRRPv3 backup device

VRRPv3 supports IPv4 sessions as well as IPv6 sessions. To configure a VRRPv3 session for IPv4, assign a virtual router group with the VRRP version set to 3. This task is performed on any device that is designated as an IPv4 backup VRRPv3 device. For each VRRP virtual routing instance, there is one master device and all other devices are backups. Repeat this task on all devices that are to be designated as backup devices.

1. On a device designated as a backup VRRP device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP.

```
device(config)# router vrrp
```

3. Configure the Ethernet interface.

```
device(config)# interface ethernet 1/5
```

4. Configure the IP address of the interface. All devices configured for the same virtual router ID (VRID) must be on the same subnet.

```
device(config-if-e1000-1/5)# ip address 10.53.5.3/24
```

5. Assign the same VRID as the VRID used by the owner device.

```
device(config-if-e1000-1/5)# ip vrrp vrid 1
```

   NOTE
   You can assign a VRID number in the range of 1 through 255.

6. Designate this router as a backup VRRP device.

```
device(config-if-e1000-1/5-vrid-1)# backup priority 110
```

   While configuring a backup device, you can set a priority that is used when a master VRRP device goes offline. The backup device with the highest priority will assume the role of master device.

7. Configure the IP address of the VRID.

```
device(config-if-e1000-1/5-vrid-1)# ip-address 10.53.5.1
```

   The VRID IP address is the same virtual IP address that you used for the VRRP owner device.

8. Enable the VRRP session.

```
device(config-if-e1000-1/5-vrid-1)# activate
VRRP router 1 for this interface is activating
```

The following example configures a VRRP owner device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/5
device(config-if-e1000-1/5)# ip address 10.53.5.3/24
device(config-if-e1000-1/5)# ip vrrp vrid 1
device(config-if-e1000-1/5-vrid-1)# backup priority 110
device(config-if-e1000-1/5-vrid-1)# ip-address 10.53.5.1
device(config-if-e1000-1/5-vrid-1)# activate
VRRP router 1 for this interface is activating
```

Tracked ports and track priority with VRRP and VRRP-E

Port tracking allows interfaces not configured for VRRP or VRRP-E to be monitored for link-state changes that can result in dynamic changes to the VRRP device priority.

A tracked port allows you to monitor the state of the interfaces on the other end of a route path. A tracked interface also allows the virtual router to lower its priority if the exit path interface goes down, allowing another virtual router in the same VRRP (or VRRP-E) group to take over. When a tracked interface returns to an up state, the configured track priority is added to the current virtual router priority value. The following conditions and limitations exist for tracked ports:

- Track priorities must be lower than VRRP or VRRP-E priorities.
- The dynamic change of router priority can trigger a master device switchover if preemption is enabled. However, if the router is an owner, the master device switchover will not occur.
- The maximum number of interfaces that can be tracked for a virtual router is 16.
- Port tracking is allowed for physical interfaces and port channels.

Port tracking using IPv6 VRRPv3

The tracking of the link status of an interface not configured for VRRP or VRRP-E can be configured with a priority that can result in dynamic changes to the VRRP device priority.

After enabling IPv6 VRRPv3 you can configure tracking the port status of other interfaces on the device that are not configured for VRRP. Any link down or up events from tracked interfaces can result in dynamic changes in the virtual router priority and a potential master device switchover. The configured priority must be less than the VRRPv3 or VRRP-Ev3 priorities.

1. Enter the **configure** command to access global configuration mode.

```
device# configure
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 125
```

3. To globally enable VRRPv3, enter the **ipv6 protocol vrrp** command.

```
device(config-rbridge-id-125)# ipv6 protocol vrrp
device(config)# ipv6 protocol vrrp
```

4. Enter the **interface ve** command with an associated VLAN number.

```
device(config-rbridge-id-125)# interface ve 2018
device(config)# interface ve 2018
```

   In this example, virtual Ethernet (ve) interface configuration mode is entered and the interface is assigned with a VLAN number of 2018.

5. Enter an IPv6 address for the interface using the **ipv6 address** command.

```
device(config-ve-2018)# ipv6 address 2001:2018:8192::125/64
```

6. Enter the **ipv6 vrrp-group** command with a number to assign a VRRPv3 group to the device.

```
device(config-ve-2018)# ipv6 vrrp-group 18
```

   In this example, VRRP group configuration mode is entered.

7. Enter the **virtual-ip** command to assign a link-local virtual IPv6 address to a VRRPv3 group.

```
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
```

   In this example, the link-local IPv6 address of the virtual router is assigned to VRRPv3 group 18. The first virtual IP address entered enables the VRRPv3 session.

8. Enter the **virtual-ip** command to assign a virtual IPv6 address to a VRRPv3 group.

```
device(config-vrrp-group-18)# virtual-ip 2001:2018:8192::1
```

   In this example, the IPv6 address of the virtual router is assigned to VRRPv3 group 18.

9. Enter the **track** command with an interface and a priority to enable the tracking of ports that are not configured as VRRP interfaces.

```
device(config-vrrp-group-18)# track tengigabitethernet 3/0/5 priority 15
device(config-vrrp-group-18)# track ethernet 1/5 priority 15
device(config-vrrp-group-18)# track ethernet 0/5 priority 15
```

10. Enter the **no preempt-mode** command to disable preemption.

```
device(config-vrrp-group-18)# no preempt-mode
```

   Preemption can be disabled when you do not want to preempt an existing master with a higher priority device.

11. Enter the **priority** command to configure the priority of the virtual router. In VRRPv3, the virtual router with the highest priority becomes the master VRRPv3 device.

```
device(config-vrrp-group-18)# priority 120
```

The following example shows how to configure an IPv6 VRRPv3 session and enable the tracking of a 10 GbE interface.

```
device# configure
device(config)# rbridge-id 125
device(config-rbridge-id-125)# ipv6 protocol vrrp
device(config-rbridge-id-125)# interface ve 2018
device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
device(config-ve-2018)# ipv6 vrrp-group 18
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
device(config-vrrp-group-18)# virtual-ip 2001:2018:8192::1
device(config-vrrp-group-18)# track tengigabitethernet 3/0/5 priority 15
device(config-vrrp-group-18)# no preempt-mode
device(config-vrrp-group-18)# priority 120

device# configure
device(config)# ipv6 protocol vrrp
device(config)# interface ve 2018
device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
device(config-ve-2018)# ipv6 vrrp-group 18
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
device(config-vrrp-group-18)# track ethernet 1/5 priority 15
device(config-vrrp-group-18)# no preempt-mode
device(config-vrrp-group-18)# priority 120

device# configure
device(config)# ipv6 protocol vrrp
device(config)# interface ve 2018
device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
device(config-ve-2018)# ipv6 vrrp-group 18
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
device(config-vrrp-group-18)# track ethernet 0/5 priority 15
device(config-vrrp-group-18)# no preempt-mode
device(config-vrrp-group-18)# priority 120
```

Tracking ports and setting VRRP priority using VRRPv3

Configuring port tracking on an exit path interface and setting a priority on a VRRPv3 device enables VRRPv3 to monitor the interface. For VRRPv3, if the interface goes down, the device priority is set to the priority value and another backup device with a higher priority assumes the role of master. For VRRP-Ev3, if the interface goes down, the device priority is lowered by the priority value and another backup device with a higher priority assumes the role of master.

Before enabling IPv6 VRRPv3, you must globally enable IPv6 routing using the **ipv6 unicast-routing** command.

Configure this task on the device on which the tracked interface exists.

> NOTE
> Only IPv4/IPv6 IPsec tunnels can be configured with a specific **track-port** command priority. Other interfaces use the track priority associated with the **owner** or **backup** commands.

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

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 router vrrp** command to configure VRRPv3 globally.

   ```
   device(config)# ipv6 router vrrp
   ```

3. Configure the Ethernet interface.

   ```
   device(config)# interface ethernet 1/6
   ```

4. Enter the IPv6 address for the interface to be used for the virtual router ID (VRID).

```
device(config-if-e10000-1/6)# ipv6 address fd2b::2/64
```

5. Enter the following command to enter the appropriate VRRPv3 virtual router ID (VRID) mode.

```
device(config-if-e10000-1/6)# ipv6 vrrp vrid 1
```

6. Enter the **track-port** command to set the tracked port and priority:

```
device(config-if-e10000-1/6-vrid-1)# track-port tunnel 1 priority 20
```

   The priority value is used when a tracked port goes down and the new priority is set to this value. Ensure that the priority value is lower than the priority set for any existing master or backup device to force a renegotiation for the master device.

The following example shows how to configure tunnel interface 1 on virtual router 1 to be tracked; if the interface fails, the IPv6 VRRPv3 priority of the device becomes 20, forcing a negotiation for a new master device.

```
device# configure terminal
device(config)# ipv6 router vrrp
device(config)# interface ethernet 1/6
device(config-if-e10000-1/6)# ipv6 address fd2b::2/64
device(config-if-e10000-1/6)# ipv6 vrrp vrid 1
device(config-if-e10000-1/6-vrid-1)# track-port tunnel 1 priority 20
```

Tracked IPsec tunnels and track priority with VRRP and VRRP-E

IPsec tunnel tracking using VRRP or VRRP Extended (VRRP-E) is helpful in monitoring network reachability changes that can result in dynamic changes to the master VRRP or VRRP-E device priority.

Business-critical network traffic requires security options to be configured in the network. VRRP or VRRP-E can be implemented together with IP security (IPsec) to provide redundancy and scalablity. The tunnels to be tracked are outgoing interfaces for the device on which they are configured. If any of the tracked tunnels goes down, VRRP or VRRP-E can quickly assign a new master device to minimize any loss of packets through the network. For VRRP, if the tracked tunnel goes down, the current router priority is reduced to the priority set for the tracked tunnel resulting in renegotiation for the master device. For VRRP-E, if the tracked tunnel goes down, the current router priority is reduced by the priority set for the tracked tunnel resulting in renegotiation for the master device. When a tracked tunnel returns to an up state, the priority of the VRRP or VRRP-E device is restored to the original value to force a renegotiation for the master device. The following conditions and limitations exist for tracked tunnels:

- Track priorities must be lower than VRRP or VRRP-E priorities.
- The dynamic change of router priority can trigger a master device switchover if preemption is enabled and if the owner router goes down.
- The maximum number of tunnels or interfaces that can be tracked for a virtual router is 8.
- IPsec tunnel tracking is supported for IPv4 VRRP and IPv4 or IPv6 VRRP-E. IPv6 VRRP does not support IPsec tunnels.
- IPv4 VRRP or VRRP-E can track only IPv4 IPsec tunnels and IPv6 VRRP-E can track only IPv6 IPsec tunnels.

  NOTE
  Tracking IPsec tunnels is supported only on BR-MLX-10GX4-M-IPSEC modules running on MLXe devices.

Configuring VRRP tracking for IPsec tunnels

IPsec tunnels can be tracked for a VRRP virtual routing instance.

This task is performed on any device configured with IPsec tunnels that is designated as a VRRP device. Before configuring IPsec tunnel tracking, IPsec tunnels must be configured. Repeat this task for all devices on which VRRP IPsec tunnel tracking is required up to a maximum of 8 tracked tunnels or interfaces.

NOTE
IPsec tunnel tracking is supported for IPv4 VRRP and IPv4 or IPv6 VRRP-E. IPv6 VRRP does not support IPsec tunnels.

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

```
device# configure terminal
```

2. Enter the **router vrrp** command to configure VRRP globally.

```
device(config)# router vrrp
```

3. Configure the Ethernet interface.

```
device(config)# interface ethernet 1/6
```

4. Enter the IP address for the interface to be used for the virtual router ID (VRID).

```
device(config-if-e10000-1/6)# ip address 10.53.5.3/24
```

5. Enter the following command to enter the appropriate VRRP virtual router ID (VRID) mode.

```
device(config-if-e10000-1/6)# ip vrrp vrid 1
```

6. Enter the **track-port** command to set the track port and priority:

```
device(config-if-e10000-1/6-vrid-1)# track-port tunnel 1 priority 20
```

The priority value is used when a tracked port goes down and the new priority is set to this value. Ensure that the priority value is lower than the priority set for any existing master or backup device to force a renegotiation for the master device.

7. Exit to Privileged EXEC mode.

```
device(config-if-e10000-1/6)# end
```

8. Verify the tracked IPsec tunnel configuration using the **show ip vrrp** command.

```
device# show ip vrrp

VRID 1 (index 1)
 interface 1/6
 state MASTER
 administrative-status disabled
 version v2
 mode incomplete
 virtual mac 0000.5e00.0101
 priority 100
 current priority 100
 track-priority 20
 hello-interval 1 sec
 backup hello-interval 60 sec
 track-port tunnel 1(up)
```

The partial output shows the tracked port tunnel number and status.

The following example shows how to configure tunnel 1 on virtual router 1 to be tracked; if the tunnel fails, the VRRP priority of the device becomes 20, forcing a negotiation for a new master device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/6
device(config-if-e10000-1/6)# ip address 10.53.5.1/24
device(config-if-e10000-1/6)# ip vrrp vrid 1
device(config-if-e10000-1/6-vrid-1)# track-port tunnel 1 priority 20
```

Configuring VRRP-E tracking for IPsec tunnels

IPsec tunnels can be tracked for a VRRP-E virtual routing instance.

This task is performed on any device that is designated as a VRRP extended (VRRP-E) device. Before configuring IPsec tunnel tracking, IPsec tunnels must be configured. Repeat this task for all devices on which VRRP-E IPsec tunnel tracking is required up to a maximum of 8 tracked tunnels or interfaces.

> **NOTE**
> IPsec tunnel tracking is supported for IPv4 VRRP and IPv4 or IPv6 VRRP-E. IPv6 VRRP does not support IPsec tunnels.

1. On the device designated as a VRRP-E device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

   ```
   device# configure terminal
   ```

2. Globally enable VRRP-E.

   ```
   device(config)# router vrrp-extended
   ```

3. Configure the Ethernet interface link.

   ```
   device(config-vrrpe-router)# interface ethernet 1/5
   ```

4. Configure the IP address of the interface. All devices configured for the same virtual router ID (VRID) must be on the same subnet.

   ```
   device(config-if-e1000-1/5)# ip address 10.53.5.3/24
   ```

5. Assign the device to VRID 1.

   ```
   device(config-if-e1000-1/5)# ip vrrp-extended vrid 1
   ```

   > **NOTE**
   > You can assign a VRID number in the range of 1 through 255.

6. Designate this router as a backup VRRP device.

   ```
   device(config-if-e1000-1/5-vrid-1)# backup priority 50 track priority 10
   ```

   While configuring a backup device, you can set a priority that is used when a master VRRP device goes offline. The backup device with the highest priority will assume the role of master device.

7. Configure the IP address of the VRID.

   ```
   device(config-if-e1000-1/5-vrid-1)# ip-address 10.53.5.254
   ```

   The IP address associated with the VRID must not be configured on any of the devices used for VRRP-E.

8. Enter the **track-port** command to set the track port and priority for one end of the IPsec tunnel.

```
device(config-if-e1000-1/5-vrid-1)# track-port tunnel 1 priority 10
```

   The priority value is used when a tracked tunnel goes down and the new priority is reduced by this value. Ensure that the priority value is lower than the priority set for any existing master or backup device to force a renegotiation for the master device.

9. Enter the **track-port** command to set the track port and priority of the IPsec tunnel.

```
device(config-if-e1000-1/5-vrid-1)# track-port tunnel 2 priority 20
```

   The priority value is used when a tracked port goes down and the new priority is reduced by this value. Ensure that the priority value is lower than the priority set for any existing master or backup device to force a renegotiation for the master device.

10. Enable the VRRP-E session.

```
device(config-if-e1000-1/5-vrid-1)# activate
VRRP-E router 1 for this interface is activating
```

The following example configures a VRRP-E device to track IPsec tunnels 1 and 2.

```
device# configure terminal
device(config)# router vrrp-extended
device(config-vrrpe-router)# interface ethernet 1/5
device(config-if-e1000-1/5)# ip address 10.53.5.3/24
device(config-if-e1000-1/5)# ip vrrp-extended vrid 1
device(config-if-e1000-1/5-vrid-1)# backup
device(config-if-e1000-1/5-vrid-1)# ip-address 10.53.5.1
device(config-if-e1000-1/5-vrid-1)# track-port tunnel 1 priority 10
device(config-if-e1000-1/5-vrid-1)# track-port tunnel 2 priority 20
device(config-if-e1000-1/5-vrid-1)# activate
VRRP-E router 1 for this interface is activating
```

VRRP hold timer

The hold timer delays the preemption of a master VRRP device by a high-priority backup device.

A hold timer is used when a VRRP-enabled device that was previously a master device failed, but is now back up. This restored device now has a higher priority than the current VRRP master device, and VRRP normally triggers an immediate switchover. In this situation, it is possible that not all software components on the backup device have converged yet. The hold timer can enforce a waiting period before the higher-priority backup device assumes the role of master VRRP device again. The timer must be set to a number greater than 0 seconds for this functionality to take effect.

Hold timer functionality is supported in both version 2 and version 3 of VRRP and VRRP-E.

Configuring VRRP hold timer support

A hold timer can be configured on a VRRP-enabled interface to set an interval, in seconds, before a backup device becomes the master VRRP device.

A hold timer is used when a VRRP-enabled device that was previously a master device failed, but is now back online. The backup device has a higher priority than the current VRRP master device. Before assuming the role of master VRRP device again, the backup device waits for the time period specified in the hold timer. This task is supported in both versions of VRRP and VRRP-E, but the configuration below is for VRRPv3.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 125
```

3. Enable IPv6 VRRP-E.

```
device(config-rbridge-id-125)# ipv6 protocol vrrp-extended
device(config)# ipv6 protocol vrrp-extended
```

In this example, virtual Ethernet (ve) interface configuration mode is entered and the interface is assigned with a VLAN number of 2018.

4. Enter the **interface ve** command with an associated vlan number.

```
device(config-rbridge-id-125)# interface ve 2018
device(config)# interface ve 2018
```

In this example, virtual Ethernet (ve) interface configuration mode is entered and the interface is assigned with a VLAN number of 2018.

5. Enter an IPv6 address for the interface using the **ipv6 address** command.

```
device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
```

6. Enter the **ipv6 vrrp-group** command with a number to assign a VRRPv3 group to the device.

```
device(config-ve-2018)# ipv6 vrrp-group 18
```

In this example, VRRP group configuration mode is entered.

7. Enter the **description** command to enter text that describes the virtual router group.

```
device(config-vrrp-group-18)# description Product Marketing group
```

8. Enter the **advertisement-interval** command with a number representing milliseconds.

```
device(config-vrrp-group-18)# advertisement-interval 3000
```

> NOTE
> In VRRPv3, the advertisement-interval is in milliseconds.

9. Enter the **hold-time** command with a number representing seconds.

```
device(config-vrrp-group-18)# hold-time 5
```

The following example configures and enables a VRRPv3 session and adds a VRRP group description. A hold time of 5 seconds is configured. This example also contains appropriate **virtual-ip** command configuration not included in the task above.

```
device# configure
device(config)# rbridge-id 125
device(config-rbridge-id-125)# ipv6 protocol vrrp-extended
device(config-rbridge-id-125)# interface ve 2018
device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
device(config-ve-2018)# ipv6 vrrp-group 18
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
device(config-vrrp-group-18)# virtual-ip 2001:2018:8192::1
device(config-vrrp-group-18)# description Product Marketing group
device(config-vrrp-group-18)# advertisement-interval 3000
device(config-vrrp-group-18)# hold-time 5

device# configure
device(config)# ipv6 protocol vrrp-extended
device(config)# interface ve 2018
device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
device(config-ve-2018)# ipv6 vrrp-group 18
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
device(config-vrrp-group-18)# virtual-ip 2001:2018:8192::1
device(config-vrrp-group-18)# description Product Marketing group
device(config-vrrp-group-18)# advertisement-interval 3000
device(config-vrrp-group-18)# hold-time 5

device# configure
device(config)# ipv6 protocol vrrp-extended
device(config)# interface ve 2018
device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
device(config-ve-2018)# ipv6 vrrp-group 18
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
device(config-vrrp-group-18)# description Product Marketing group
device(config-vrrp-group-18)# advertisement-interval 3000
device(config-vrrp-group-18)# hold-time 5
```

Accept mode for backup VRRP devices

Accept mode allows a backup VRRP device to respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

For each VRRP virtual routing instance, there is one master device and all other devices are backups. Accept mode allows some network management functionality for backup VRRP devices, providing the ability to respond to ping, traceroute, and Telnet packets. Troubleshooting network connections to the VRRP nonowner master device is difficult unless accept mode is enabled. By default, accept mode is enabled and non-owner VRRP devices will accept packets destined for the IPv4 or IPv6 VRID addresses if they become master.

> NOTE
> The accept mode functionality enables a VRRP nonowner master device to respond to ping, Telnet, and traceroute packets, but the device will not respond to SSH packets.

Disabling accept mode on a backup VRRP device

When accept mode is disabled on a backup VRRP device, it will not respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

Perform this task on any device that is designated as a backup VRRP device. The functionality is activated if the backup device becomes a master VRRP device. Repeat this task for all devices that are to be designated as backup devices.

NOTE
The accept mode functionality does not support SSH packets.

1. On the device designated as a backup VRRP device, from privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Disable VRRP accept mode for all instances of VRRP on the device.

```
device(config)# vrrp-acceptmode-disable
```

You can re-enable accept mode by using the **no** form of this command.

Accept mode for backup VRRP devices

Accept mode allows a backup VRRP device to respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

For each VRRP virtual routing instance, there is one master device and all other devices are backups. Accept mode allows some network management functionality for backup VRRP devices, providing the ability to respond to ping, traceroute, and Telnet packets. By default, nonowner VRRP devices do not accept packets destined for the IPv4 or IPv6 VRID addresses. Troubleshooting network connections to the VRRP nonowner master device is difficult unless accept mode is enabled.

NOTE
The accept mode functionality enables a VRRP nonowner master device to respond to ping, Telnet, and traceroute packets, but the device will not respond to SSH packets.

Enabling accept mode on a backup VRRP device

Enabling accept mode allows a backup VRRP device to respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

This task is performed on any device that is designated as a backup VRRP device, and the functionality is activated if the backup device becomes a master VRRP device. Repeat this task for all devices that are to be designated as backup devices.

NOTE
The accept mode functionality does not support SSH packets.

1. On the device designated as a backup VRRP device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP.

```
device(config)# router vrrp
```

3. Configure the Ethernet interface link.

```
device(config)# interface ethernet 1/1/5
```

4. Configure the IP address of the interface. All devices configured for the same virtual router ID (VRID) must be on the same subnet.

```
device(conf-if-e1000-1/1/5)# ip address 10.53.5.3/24
```

5. Assign this backup device to VRID 1, the same VRID as the VRRP owner device.

```
device(conf-if-e1000-1/1/5)# ip vrrp vrid 1
```

NOTE
You can assign a VRID number in the range of 1 through 255.

6. Designate this router as a backup VRRP device.

```
device(conf-if-e1000-1/1/5-vrid-1)# backup priority 110
```

While configuring a backup device, you can set a priority that is used when a master VRRP device goes offline. The backup device with the highest priority will assume the role of master device.

7. Enable accept mode for this device.

```
device(conf-if-e1000-1/1/5-vrid-1)# accept-mode
```

8. Exit configuration mode and return to privileged EXEC mode.

```
device(conf-if-e1000-1/1/5-vrid-1)# end
```

9. Verify that accept mode is enabled.

```
device# show ip vrrp vrid 1

Interface 1/1/5
----------------
auth-type no authentication
VRID 1 (index 1)
 interface 1/1/5
 state master
 administrative-status enabled
 version v2
 mode non-owner (backup)
 virtual mac aaaa.bbbb.cccc (configured)
 priority 110
 current priority 110
 track-priority 2
 hello-interval 1 sec
 accept-mdoe enabled
.
.
.
```

The following example enables accept mode for a backup VRRP device.

Accept mode for backup VRRP devices

Accept mode allows a backup VRRP device to respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

For each VRRP virtual routing instance, there is one master device and all other devices are backups. Accept mode allows some network management functionality for backup VRRP devices, providing the ability to respond to ping, traceroute, and Telnet packets. Troubleshooting network connections to the VRRP non-owner master device is difficult unless accept mode is enabled. By default, accept mode is enabled and non-owner VRRP devices will accept packets destined for the IPv4 or IPv6 VRID addresses if they become master.

> NOTE
> The accept mode functionality enables a VRRP non-owner master device to respond to ping, Telnet, and traceroute packets, but the device will not respond to SSH packets.

Disabling accept mode on a backup VRRP device

When accept mode is disabled on a backup VRRP device, it will not respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

Perform this task on any device that is designated as a backup VRRP device. The functionality is activated if the backup device becomes a master VRRP device. Repeat this task for all devices that are to be designated as backup devices.

> NOTE
> The accept mode functionality does not support SSH packets.

1. On the device designated as a backup VRRP device, from privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Run the **rbridge-id** command with the RBridge ID.

   ```
   device(config)# rbridge-id 1
   ```

3. Disable VRRP accept mode for all instances of VRRP on the device.

   ```
   device(config-rbridge-id-1)# vrrp-acceptmode-disable
   ```

   You can use the **no** form of this command to re-enable accept mode.

Alternate VRRPv2 checksum for VRRPv3 IPv4 sessions

If VRRPv3 is configured on an Extreme device in a network with third-party peering devices using VRRPv2-style checksum calculations for IPv4 VRRPv3 sessions, a VRRPv2-style checksum must be configured for VRRPv3 IPv4 sessions on the device.

VRRPv3 introduced a new checksum method for both IPv4 and IPv6 sessions, and this version 3 checksum computation is enabled by default. To accommodate third-party devices that still use a VRRPv2-style checksum for IPv4 VRRPv3 sessions, a command-line interface (CLI) command is available for configuration on a device. The new version 2 checksum method is disabled by default and is applicable only to IPv4 VRRPv3 sessions. If configured for VRRPv2 sessions, the VRRPv2-style checksum command is accepted, but it has no effect.

Enabling the v2 checksum computation method in a VRRPv3 IPv4 session

Enabling the alternate VRRPv2-style checksum in a VRRPv3 IPv4 session for compatibility with third-party network devices.

VRRPv3 uses the v3 checksum computation method by default for both IPv4 and IPv6 sessions on this device. Third-party devices may only have a VRRPv2-style checksum computation available for a VRRPv3 IPv4 session. The **use-v2-checksum** command is entered in interface configuration mode.

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

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 125
   ```

3. To enable VRRP globally enter the **protocol vrrp** command.

   ```
   device(config-rbridge-id-125)# protocol vrrp
   device(config)# protocol vrrp
   ```

4. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config-rbridge-id-125)# interface ve 2018
   device(config)# interface ve 2018
   ```

5. To assign an IPv4 VRRPv3 group to the device use the **vrrp-group** command with a group number and version 3.

   ```
   device(config-ve-2018)# vrrp-group 10 version 3
   ```

6. To enable v2 checksum computation method in an IPv4 VRRPv3 session, use the **use-v2-checksum** command in the VRRP group configuration mode.

   ```
   device(config-vrrp-group-10)# use-v2-checksum
   ```

The following example shows the v2 checksum computation method enabled for an VRRPv3 IPv4 session on a device.

```
device# configure terminal
device(config)# rbridge-id 125
device(config-rbridge-id-125)# protocol vrrp
device(config-rbridge-id-125)# interface ve 2018
device(config-ve-2018)# vrrp-group 10 version 3
device(config-vrrp-group-10)# use-v2-checksum

device# configure terminal
device(config)# protocol vrrp
device(config)# interface ve 2018
device(config-ve-2018)# vrrp-group 10 version 3
device(config-vrrp-group-10)# use-v2-checksum
```

Enabling the VRRPv2 checksum computation method in a VRRPv3 IPv4 session

An alternate VRRPv2-style checksum can be configured in a VRRPv3 IPv4 session for compatibility with third-party network devices.

VRRPv3 uses the v3 checksum computation method by default for both IPv4 and IPv6 sessions on the NetIron OS devices. Third-party devices may have only a VRRPv2-style checksum computation available for a VRRPv3 IPv4 session. The **use-v2-checksum** command is entered in interface configuration mode.

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

   ```
   device# configure terminal
   ```

2. Enable VRRP globally.

   ```
   device(config)# router vrrp
   ```

3. Enter the **interface** command with an interface type and number.

   ```
   device(config)# interface ethernet 2/4
   ```

4. To configure a VRRP virtual routing ID, use the **ip vrrp vrid** command with an associated ID number.

   ```
   device(config-if-e1000-2/4)# ip vrrp vrid 14
   ```

5. To enable VRRP version 3 (VRRPv3), enter the **version** command with a version number of v3.

   ```
   device(config-if-e1000-2/4-vrid-14)# version v3
   ```

6. To enable the v2 checksum computation method in an IPv4 VRRPv3 session, use the **use-v2-checksum** command in VRRP configuration mode.

   ```
   device(config-if-e1000-2/4-vrid-14)# use-v2-checksum
   ```

7. Enter the IP address for the interface using the **ip-address** command.

   ```
   device(config-if-e1000-2/4-vrid-14)# ip-address 10.14.14.99
   ```

8. To activate the interface, enter the **activate** command.

   ```
   device(config-if-e1000-2/4-vrid-14)# activate
   ```

The following example shows the v2 checksum computation method enabled for an VRRPv3 IPv4 session on a device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 2/4
device(config-if-e1000-2/4)# ip vrrp vrid 14
device(config-if-e1000-2/4-vrid-14)# version v3
device(config-if-e1000-2/4-vrid-14)# use-v2-checksum
device(config-if-e1000-2/4-vrid-14)# ip-address 10.14.14.99
device(config-if-e1000-2/4-vrid-14)# activate
```

Displaying alternate VRRPv2 checksum settings

The verification of the use of the alternate VRRPv2-style checksum for VRRPv3 IPv4 sessions is achieved using several CLI commands.

The following steps are both optional and can be used to verify that the alternate VRRPv2-style checksum computation command, **use-v2-checksum**, has been set for VRRPv3 IPv4 sessions.

1. Use the **show running-config** command to verify that the **use-v2-checksum** command has been configured for a specified interface. Only part of the output is displayed.

```
device# show running-config

interface ethernet 2/4
 ip address 10.14.14.2/24
 ip vrrp vrid 14
  backup
  ip-address 10.14.14.99
  use-v2-checksum
  exit
```

2. Use the **show ip vrrp** command with a virtual router ID number to display the current settings of a specific VRRP session, including the **use-v2-checksum** command, if configured.

```
device# show ip vrrp vrid 14

Interface 2/4
----------------
auth-type no authentication

VRID 14 (index 1)
 interface 2/4
 state initialize
 administrative-status disabled
 version v3 - use-v2-checksum
 mode non-owner(backup)
 virtual mac 0000.5e00.010e
 priority 100
 current priority 100
 track-priority 1
 hello-interval 1 sec
 backup hello-interval 60 sec
 slow-start timer (configured) 0 sec
 advertise backup disabled
 dead-interval 3500 ms
 preempt-mode true
 ip-address 10.14.14.99
```

VRRPv3 router advertisement suppression

VRRPv3 introduces the ability to suppress router advertisements (RAs).

Router advertisements are sent by the VRRP master device and contain the link-local virtual IP address and the virtual MAC address. For network security reasons, if you do not want the MAC addresses of interfaces to be viewed, you can disable RA messages. Disabling RA does not remove the auto-configured addresses being sent by VRRP updates, but the RA messages are dropped by the router interface. There are two other situations where you may want to disable RA messages:

- If an interface is currently the VRRP master but the virtual IP address is not the address of this interface, the device should not send RA messages for the interface IP address.
- If the interface is in a backup state, the device should not send RA messages for the interface IP address.

Disabling VRRPv3 router advertisements

The ability to suppress VRRPv3 master device interface router advertisements is introduced.

Suppressing interface router advertisements from the master VRRPv3 device may be performed for network security concerns because the RA messages include the MAC addresses of interfaces. In this task, VRRP-Ev3 is configured globally and RA messages are suppressed for the virtual ethernet (VE) 2109 interface.

> NOTE
> To configure this task for VRRPv3, use the **ipv6 protocol vrrp** command.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Globally enable VRRP-Ev3.

   ```
   device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
   device(config)# ipv6 protocol vrrp-extended
   ```

4. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config-rbridge-id-122)# interface ve 2019
   device(config)# interface ve 2019
   ```

   In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned with a VLAN number of 2019.

5. Enter the **ipv6 vrrp-suppress-interface-ra** command to suppress interface RA messages for the ve 2019 interface.

   ```
   device(config-ve-2019)# ipv6 vrrp-suppress-interface-ra
   ```

The following example shows how to disable VRRPv3 RA messages from interface configuration mode for a VRRP-Ev3 session.

```
device# configure
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
device(config-rbridge-id-122)# interface ve 2019
device(config-ve-2019)# ipv6 vrrp-suppress-interface-ra

device# configure
device(config)# ipv6 protocol vrrp-extended
device(config)# interface ve 2019
device(config-ve-2019)# ipv6 vrrp-suppress-interface-ra
```

Automatic generation of a virtual link-local address for VRRPv3

The virtual MAC address is used to automatically generate the IPv6 virtual link-local address to simplify the configuration of IPv6 VRRP and standardize implementations across vendor platforms. Subsequent VRRPv3 advertisements carry the auto-generated virtual link-local address.

The default VRRPv3 implementation allows only the link-local address that is configured on a physical interface to be used as the virtual IPv6 address of a VRRPv3 session. This limits configuring a link-local address for each VRRP instance on the same physical interface because there can be only one link-local address per physical interface.

When IPv6 link-local address auto-generation is configured for IPv6 VRRP, a virtual IPv6 link-local address is generated automatically using the EUI-64 result of the virtual MAC address. The virtual IPv6 link-local address is generated for a specific VRRP instance and the virtual link-local address is carried in VRRPv3 advertisements. The auto-generation process is defined in RFC 5798 allowing cross-vendor platform support. This ability to generate a link-local address automatically depends on the existence of a consistent virtual MAC address in the local network.

If the virtual link-local address is configured manually, the configured address takes precedence over the automatically generated address. The administrator should ensure that the configured virtual link-local address is consistent across all routers in the LAN. When the manually configured address is removed, the auto-generated address is used.

If there is a mismatch in the IPv6 addresses field, the devices drop the advertisements that are sent by backup VRRP routers. The advertisements from the master VRRP router are not dropped regardless of the IPv6 address comparison. The virtual MAC must be consistent on the local network. When the virtual MAC is modified, the virtual link-local address is regenerated.

As an Extreme proprietary protocol, VRRP Extended version 3 (VRRP-Ev3) is not supported.

Enabling auto-generation of an IPv6 virtual link-local address

This task is performed on all the devices that are designated as backup VRRPv3 devices. Initially a backup priority is set to 100. For each VRRPv3 session there are master and backup routers, the IPv6 address assigned here to the VRID is the IPv6 address of the master router. The task is repeated on each backup VRRPv3 device with corresponding changes to the interface number and IPv6 address of the interface.

1. On the device designated as a backup VRRPv3 device, from privileged EXEC mode, enter configuration mode by issuing the **configure terminal** command.

   ```
   device# configure terminal
   ```

2. Globally enable the IPv6 VRRP protocol.

   ```
   device(config)# ipv6 router vrrp
   ```

3. Configure the ethernet interface link for the owner device.

   ```
   device(config)# interface ethernet 1/4
   ```

4. Configure the IPv6 address of the interface.

   ```
   device(config-if-e1000-1/4)# ipv6 address 3013::3/64
   ```

5. Assign the backup device to the virtual router ID (VRID) 2.

```
device(config-if-e1000-1/4)# ipv6 vrrp vrid 2
```

NOTE
You can assign a VRID number in the range of 1 through 255.

6. Designate this router as the VRRPv3 backup device and assign a priority of 100.

```
device(config-if-e1000-1/4-vrid-1)# backup priority 100
```

7. Automatically generate the IPv6 link-local address for the VRID for use in the local network.

```
device(config-if-e1000-1/4-vrid-2)# ipv6-address auto-gen-link-local
```

8. Assign the global IPv6 address to the VRID.

```
device(config-if-e1000-1/4-vrid-2)# ipv6-address 3013::2
```

9. Enable the VRRP session.

```
device(config-if-e1000-1/4-vrid-2)# activate
```

10. To verify that link-local addresses are being automatically generated, enter the **show ipv6 vrrp** command for this VRID.

```
device(config-if-e1000-1/4-vrid-2)# show ipv6 vrrp vrid 2

VRID 2 (index 1)
 interface 1/1
 state master
 administrative-status enabled
 version v3
 mode owner
 virtual mac 0000.5e00.0101
 virtual link-local fe80::200:5eff:fe00:201
 priority 255
 current priority 255
 track-priority 2
 hello-interval 1000 ms
 backup hello-interval 60000 ms
 number of configured virtual address 2
 ipv6-address 1:2:45::2
 ipv6-address 1:2:46::2
 next hello sent in 300 ms
 Track MCT-VPLS-State: Disable
```

The following example configures an automatic generation of an IPv6 VRRPv3 link-local address for backup device.

```
device# configure terminal
device(config)# ipv6 router vrrp
device(config-ipv6-vrrp-router)# interface ethernet 1/4
device(config-if-e1000-1/4)# ipv6 address 3013::3/64
device(config-if-e1000-1/4)# ipv6 vrrp vrid 2
device(config-if-e1000-1/4-vrid-2)# backup priority 100
device(config-if-e1000-1/4-vrid-2)# ipv6-address auto-gen-link-local
device(config-if-e1000-1/4-vrid-2)# ipv6-address 3013::2
device(config-if-e1000-1/4-vrid-2)# activate
```

Displaying VRRPv3 statistics

Various show commands can display statistical information about IPv6 VRRP configurations.

Before displaying statistics, VRRPv3 must be configured and enabled in your network to generate traffic.

Use one or more of the following commands to display VRRPv3 information. The commands do not have to be entered in this order.

1. Use the **exit** command to return to privileged EXEC mode, if required.
2. Enter the **show ipv6 vrrp summary** command.

```
device# show ipv6 vrrp summary

============Rbridge-id:122============

Total number of VRRP session(s)   : 2
Master session count  : 1
Backup session count  : 1
Init session count    : 0

VRID  Session  Interface  Admin    Current   State   Short-path  Revert    SPF
                         State    Priority          Forwarding  Priority  Reverted
====  =======  =========  =====    ========  =====   ==========  ========  ========
18    VRRPE    Ve 2018    Enabled  254       Master  Enabled     unset     No
19    VRRPE    Ve 2019    Enabled  100       Backup  Enabled     unset     No
```

   This example shows summary output for the two IPv6 VRRP-E sessions that are configured for virtual routers 18 and 19.

3. Enter the **show ipv6 vrrp summary** command.

```
device# show ipv6 vrrp summary

Total number of VRRP session(s)   : 2
Master session count  : 1
Backup session count  : 1
Init session count    : 0

VRID  Session  Interface  Admin    Current   State   Short-path  Revert    SPF
                         State    Priority          Forwarding  Priority  Reverted
====  =======  =========  =====    ========  =====   ==========  ========  ========
18    VRRPE    Ve 2018    Enabled  254       Master  Enabled     unset     No
19    VRRPE    Ve 2019    Enabled  100       Backup  Enabled     unset     No
```

   This example shows summary output for the two IPv6 VRRP-E sessions that are configured for virtual routers 18 and 19.

4. To display detailed information for a single VRRP virtual router ID(VRID), enter the **show ipv6 vrrp** command with the **detail** keyword and a specific VRID.

```
device# show ipv6 vrrp 19 detail

============Rbridge-id:122============

Total number of VRRP session(s)   : 1

VRID 19
  Interface: Ve 2019;  Ifindex: 1207961571
  Mode: VRRPE
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 3
  Authentication type: No Authentication
  State: Backup
  Session Master IP Address: fe80::205:33ff:fe79:fb1e
  Virtual IP(s): 2001:2019:8192::1
  Virtual MAC Address: 02e0.5200.2513
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: DISABLE  (default: DISABLED)
  Advertise-backup: ENABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Enabled
  Revert-Priority: unset; SPF Reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 103259, Tx: 1721
    Neighbor Advertisements             : Tx: 0
    Session becoming master    : 0
    Advts with wrong interval  : 0
    Prio Zero pkts             : Rx: 0, Tx: 0
    Invalid Pkts Rvcd          : 0
    Bad Virtual-IP Pkts        : 0
    Invalid Authenticaton type : 0
    Invalid TTL Value          : 0
    Invalid Packet Length      : 0
    VRRPE backup advt sent     : 1721
    VRRPE backup advt recvd    : 0

device# show ipv6 vrrp 19 detail

Total number of VRRP session(s)   : 1

VRID 19
  Interface: Ve 2019;  Ifindex: 1207961571
  Mode: VRRPE
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 3
  Authentication type: No Authentication
  State: Backup
  Session Master IP Address: fe80::205:33ff:fe79:fb1e
  Virtual IP(s): 2001:2019:8192::1
  Virtual MAC Address: 02e0.5200.2513
```

```
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: DISABLE  (default: DISABLED)
  Advertise-backup: ENABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Enabled
  Revert-Priority: unset; SPF Reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 103259, Tx: 1721
    Neighbor Advertisements              : Tx: 0
    Session becoming master    : 0
    Advts with wrong interval  : 0
    Prio Zero pkts             : Rx: 0, Tx: 0
    Invalid Pkts Rvcd          : 0
    Bad Virtual-IP Pkts        : 0
    Invalid Authenticaton type : 0
    Invalid TTL Value          : 0
    Invalid Packet Length      : 0
    VRRPE backup advt sent     : 1721
    VRRPE backup advt recvd    : 0
```

This example shows detailed output for the IPv6 VRRP-E session for virtual router 19.

5. Enter the **show ipv6 vrrp** command to display IPv6 VRRPv3 configuration information.

```
device(config)# show ipv6 vrrp

Total number of VRRP routers defined: 1
Interface 1/3
----------------
auth-type no authentication
VRID 13 (index 2)
interface 1/3
state master
administrative-status enabled
version v3
mode non-owner(backup)
virtual mac 0000.5e00.0217
priority 100
current priority 100
track-priority 1
hello-interval 1000 ms
backup hello-interval 60000 ms
advertise backup disabled
dead-interval 3000 ms
preempt-mode true
ipv6-address fd2b::1
next hello sent in 700 ms
short-path-forwarding disabled
```

6. To view detailed statistical information about IPv6 VRRPv3, enter the **show ipv6 vrrp statistics** command.

```
device# show ipv6 vrrp statistics

Global IPv6 VRRP statistics
-------------------------------
- received vrrp packets with checksum errors = 0
- received vrrp packets with invalid version number = 0
- received vrrp packets with unknown or inactive vrid = 0
Interface 1/3
----------------
VRID 13
- number of transitions to backup state = 1
- number of transitions to master state = 1
- total number of vrrp packets received = 0
. received backup advertisements = 19
. received packets with zero priority = 0
. received packets with invalid type = 0
. received packets with invalid authentication type = 0
. received packets with authentication type mismatch = 0
. received packets with authentication failures = 0
. received packets dropped by owner = 0
. received packets with ttl errors = 0
. received packets with ipv6 address mismatch = 0
. received packets with advertisement interval mismatch = 0
. received packets with invalid length = 0
- total number of vrrp packets sent = 1175
. sent backup advertisements = 0
. sent packets with zero priority = 0
- received neighbor solicitation packets dropped = 0
- received proxy neighbor solicitation packets dropped = 0
- received ipv6 packets dropped = 0
```

Clearing VRRPv3 statistics

VRRPv3 session counters can be cleared by using a CLI command.

Ensure that VRRPv3 is configured and enabled in your network.

An excerpt of the appropriate VRRP show command output is shown before and after the appropriate VRRP clear command is entered to demonstrate that counters are cleared.

1. Enter the **end** command, if required, to return to privileged EXEC mode.

2. Enter the **show ipv6 vrrp detail** as in the following example.

```
device# show ipv6 vrrp detail

================================================================================
Rbridge-id:14
================================================================================

Total number of VRRP session(s)   : 1

VRID 11
  Interface: Ve 100;  Ifindex: 1207959652
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 3
  Authentication type: No Authentication
  State: Master
  Session Master IP Address: Local
  Backup Router(s): fe80::205:33ff:fe65:9d44
  Virtual IP(s): fe80::1
  Virtual MAC Address: 02e0.5200.250b
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: ENABLE  (default: DISABLED)
  Advertise-backup: ENABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Enabled
  Revert-Priority: unset; SPF Reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 211, Tx: 5111
    Neighbor Advertisements             : Tx: 2556
    Session becoming master    : 1
    Advts with wrong interval  : 0
    Prio Zero pkts             : Rx: 0, Tx: 0
    Invalid Pkts Rvcd          : 0
    Bad Virtual-IP Pkts        : 0
    Invalid Authenticaton type : 0
    Invalid TTL Value          : 0
    Invalid Packet Length      : 0
    VRRPE backup advt sent     : 3
    VRRPE backup advt recvd    : 84
```

3. Enter the **clear ipv6 vrrp statistics** command.

```
device# clear ipv6 vrrp statistics
```

4. Enter the **show ipv6 vrrp detail** command to confirm the changes.

```
device# show ipv6 vrrp detail

================================================================================
Rbridge-id:14
================================================================================

Total number of VRRP session(s)   : 1

VRID 11
  Interface: Ve 100;  Ifindex: 1207959652
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 3
  Authentication type: No Authentication
  State: Master
  Session Master IP Address: Local
  Backup Router(s): fe80::205:33ff:fe65:9d44
  Virtual IP(s): fe80::1
  Virtual MAC Address: 02e0.5200.250b
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: ENABLE  (default: DISABLED)
  Advertise-backup: ENABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Enabled
  Revert-Priority: unset; SPF Reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 0, Tx: 3
    Neighbor Advertisements             : Tx: 1
    Session becoming master    : 0
    Advts with wrong interval  : 0
    Prio Zero pkts             : Rx: 0, Tx: 0
    Invalid Pkts Rvcd          : 0
    Bad Virtual-IP Pkts        : 0
    Invalid Authenticaton type : 0
    Invalid TTL Value          : 0
    Invalid Packet Length      : 0
    VRRPE backup advt sent     : 0
    VRRPE backup advt recvd    : 0
```

In this show output you can see that the statistical counters have been reset. Although some of the counters show numbers because IPv6 VRRP traffic is still flowing, the numbers are much lower than in the initial **show** command output.

5. Enter the **clear ipv6 vrrp statistics** command.

```
device# clear ipv6 vrrp statistics
```

VRRP-Ev3 Overview

VRRP Extended version 3 (VRRP-Ev3) introduces IPv6 address support to the Extreme Networks proprietary VRRP Extended version 2 (VRRP-Ev2) protocol. VRRP-Ev3 is designed to avoid the limitations in the standards-based VRRPv3 protocol.

To create VRRP-Ev3, Extreme Networks has implemented the following differences from the RFC 5798 that describes VRRPv3 to provide extended functionality and ease of configuration:

- VRRP-Ev3 does not include the concept of an owner device and a master VRRP-Ev3 device is determined by the priority configured on the device.
- While the VRRP-Ev3 virtual router IP address must belong in the same subnet as a real IP address assigned to a physical interface of the device on which VRRP-Ev3 is configured, it must not be the same as any of the actual IP addresses on any interface.
- Configuring VRRP-Ev3 uses the same task steps for all devices; no differences between master and backup device configuration. The device configured with the highest priority assumes the master role.

  NOTE
  VRRP-Ev3 is supported on the devices described in this guide. In a mixed-device environment, consult your documentation for the other devices to determine if VRRP-Ev3 is supported.
  VRRP-Ev3 does not interoperate with with VRRPv2 or VRRPv3 sessions.

Enabling IPv6 VRRP-Ev3

IPv6 VRRP-Ev3 is enabled on a device when a virtual IPv6 address is assigned to a VRRP-Ev3 group.

Before assigning a virtual IPv6 address to an IPv6 VRRPv3 group, you must configure IPv6 VRRP-Ev3 on a virtual ethernet interface and assign a VRRPv3 group to the device. The IPv6 VRRP-Ev3 session is enabled after the configuration of an IPv6 virtual IP address. The configuration example following after the individual steps represents all the steps together in order.

1. Enter the **configure** command to access the global configuration mode.

   ```
   device# configure
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. To globally enable VRRP-Ev3, enter the **ipv6 protocol vrrp-extended** command.

   ```
   device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
   device(config)# ipv6 protocol vrrp-extended
   ```

4. Enter the **interface ve** command with an associated virtual Ethernet (VE) interface number.

   ```
   device(config-rbridge-id-122)# interface ve 2019
   device(config)# interface ve 2019
   ```

   In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned with a VE number of 2019.

5. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-if-ve-2019)# ipv6 address 2001:2019:8192::122/64
   ```

6. Enter the **ipv6 vrrp-extended-group** command with a number to assign a VRRP-E group to the device.

   ```
   device(config-if-ve-2018)# ipv6 vrrp-extended-group 19
   ```

   In this example, VRRP-Ev3 group configuration mode is entered.

7. Enter the **virtual-ip** command to assign a link-local virtual IPv6 address to a VRRPv3 group.

   ```
   device(config-vrrp-extended-group-19)# virtual-ip fe80::2019:1
   ```

   In this example, the IPv6 address of the virtual router is assigned to VRRP-Ev3 group 19 and the VRRP-Ev3 session is enabled.

   > **NOTE**
   > A maximum of two virtual IPv6 addresses can be configured on VRRP-Ev3 group. For VRRPv3, Extreme recommends using two IPv6 addresses; one link local address and one global address.

8. Enter the **virtual-ip** command to assign a virtual IPv6 address to a VRRPv3 group.

   ```
   device(config-vrrp-extended-group-19)# virtual-ip 2001:2019:8192::1
   ```

   In this example, a global IPv6 address is configured for the virtual router.

The following example shows how to enable a VRRP-E-v3 session by assigning a virtual IP address to an extended VRRP-E-v3 virtual group.

```
device# configure
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
device(config-rbridge-id-122)# interface ve 2019
device(config-ve-2019)# ipv6 address 2001:2019:8192::122/64
device(config-ve-2019)# ipv6 vrrp-extended-group 19
device(config-vrrp-extended-group-19)# virtual-ip fe80::2019:1
device(config-vrrp-extended-group-19)# virtual-ip 2001:2019:8192::1
```

```
device# configure
device(config)# ipv6 protocol vrrp-extended
device(config)# interface ve 2019
device(config-if-ve-2019)# ipv6 address 2001:2019:8192::122/64
device(config-if-ve-2019)# ipv6 vrrp-extended-group 19
device(config-vrrp-extended-group-19)# virtual-ip fe80::2019:1
device(config-vrrp-extended-group-19)# virtual-ip 2001:2019:8192::1
```

After enabling a VRRP-Ev3 session, you may need to configure some optional parameters such as short-path forwarding for load-balancing or tracking an interface.

Enabling an IPv6 VRRP-Ev3 device

This task is performed on any device that is designated as a VRRP extended version 3 (VRRP-Ev3) device. For each VRRP-Ev3 virtual routing instance, there is one master device and all other devices are backups; but, unlike VRRPv3, every device is configured as a backup and the device with the highest priority becomes the master device. Repeat this task for all devices that are to be designated as VRRP-Ev3 devices.

NOTE
Only VRRPv3 or VRRP-Ev3 can be enabled in your network.

1. On the device designated as a VRRP-Ev3 device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP-Ev3.

```
device(config)# ipv6 router vrrp-extended
```

3. Configure the Ethernet interface link.

```
device(config-ipv6-vrrpe-router)# interface ethernet 1/7
```

4. Configure the IPv6 address of the interface. All devices configured for the same virtual router ID (VRID) must be on the same subnet.

```
device(config-if-e1000-1/7)# ipv6 address fd4b::4/64
```

5. Assign the device to VRID 4.

```
device(config-if-e1000-1/7)# ipv6 vrrp-extended vrid 4
```

NOTE
You can assign a VRID number in the range of 1 through 255.

6. Designate this router as a backup VRRPv3 device. All VRRP-Ev3 devices are initially configured as backup devices; the device with the highest priority assumes the role of master device.

```
device(config-if-e1000-1/7-vrid-4)# backup priority 110
```

While configuring a backup device, you can set a priority that is used when the designated master VRRP device goes offline. The backup device with the highest priority will assume the role of master device.

7. Configure an IPv6 link-local address for the VRID.

```
device(config-if-e1000-1/7-vrid-4)# ipv6-address fe80::768e:f8ff:fe2a:0089
```

8. Configure a global IPv6 address for the VRID.

```
device(config-if-e1000-1/7-vrid-4)# ipv6-address fd4b::99
```

The IPv6 address associated with the VRID must not be configured on any of the devices used for VRRP-Ev3.

9. Enable the VRRP session.

```
device(config-if-e1000-1/7-vrid-4)# activate
VRRP-E router 4 for this interface is activating
```

The following example configures a backup VRRP-Ev3 device.

```
device# configure terminal
device(config)# ipv6 router vrrp-extended
device(config-ipv6-vrrpe-router)# interface ethernet 1/7
device(config-if-e1000-1/7)# ipv6 address fd4b::4/64
device(config-if-e1000-1/7)# ipv6 vrrp-extended vrid 4
device(config-if-e1000-1/7-vrid-4)# backup  priority 50
device(config-if-e1000-1/7-vrid-4)# ipv6-address fe80::768e:f8ff:fe2a:0089
device(config-if-e1000-1/7-vrid-4)# ipv6-address fd4b::99
device(config-if-e1000-1/7-vrid-4)# activate
VRRP-E router 4 for this interface is activating
```

Configuring MD5 authentication on IPv6 VRRP-Ev3 interfaces

Interfaces can be configured with an MD5 encrypted password for authentication, and VRRP-Ev3 can use the same authentication type associated with the interfaces on which you define the virtual router.

VRRP Extended version 3 (VRRP-Ev3) must be configured on the device and the interface associated with a virtual router group.

Any VRRP-Ev3 packets that do not contain the password are dropped. If your interfaces do not use authentication, neither does VRRP-Ev3. Repeat this task on all interfaces on all devices that support the same virtual router group.

> **NOTE**
> VRRP-E is supported on the devices described in this guide. In a mixed-device environment, consult your documentation for the other devices to determine if VRRP-E is supported.

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Globally enable IPv6 VRRP-Ev3.

```
device(config)# ipv6 protocol vrrp-extended
```

3. Configure the Virtual Ethernet (VE) interface link for the VRRP-E device.

```
device(config)# interface ve 20
```

   Only ve interfaces are supported by VRRP-E.

4. Enter the MD5 password configuration using the **ipv6 vrrp-extended auth-type** command with a text password. The password will be encrypted when saved in the configuration file.

```
device(config-if-Ve-20)# ipv6 vrrp-extended auth-type md5-auth kfhb61qp
```

   When an MD5 authentication password is configured on an interface, a syslog message is displayed.

5. Exit to privileged EXEC mode.

```
device(config-if-Ve-20)# end
```

6. Display the VRRP-Ev3 configuration. In this example, only partial output is displayed to verify that MD5 authentication is configured.

```
device# show ipv6 vrrp

Total number of VRRP session(s)   : 1

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRPE
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 2
  Authentication type: MD5 Authentication
.
.
.
```

The following example configures MD5 authentication for the specified VRRP-E interface.

```
device# configure terminal
device(config)# ipv6 protocol vrrp-extended
device(config)# interface ve 20
device(config-if-Ve-20)# ipv6 vrrp-extended auth-type md5-auth kfhb61qp
device(config-if-Ve-20)# end
device# show ipv6 vrrp
```

VRRP-E load-balancing using short-path forwarding

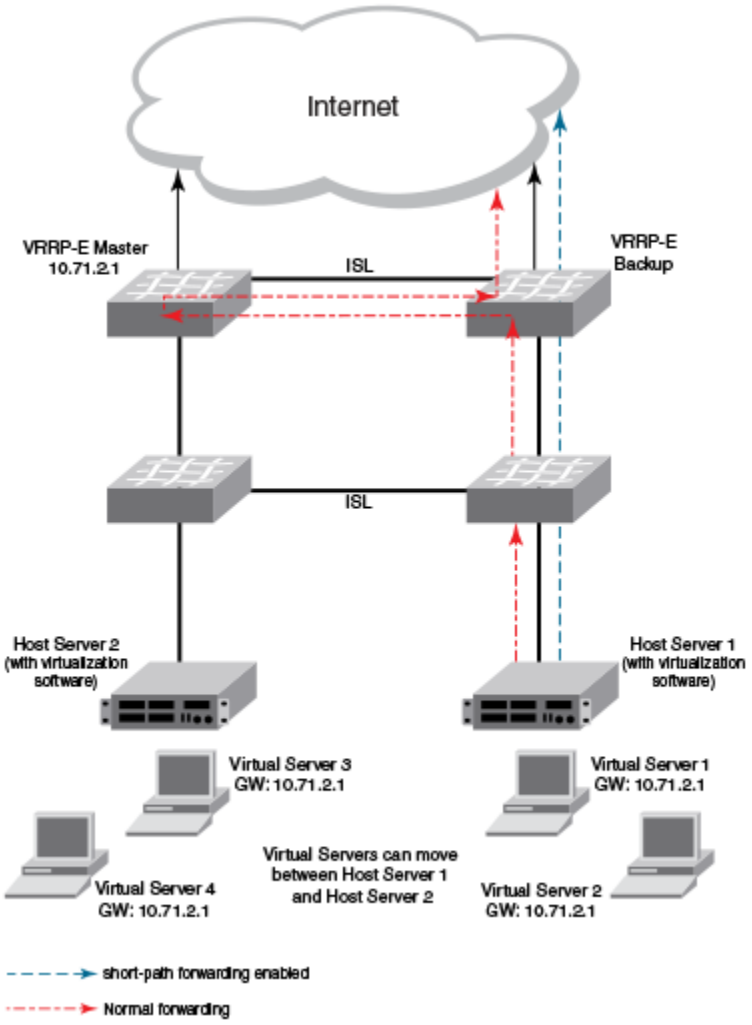
The VRRP-E Extension for Server Virtualization feature allows devices to bypass the VRRP-E master router and directly forward packets to their destination through interfaces on the VRRP-E backup router. This is called *short-path forwarding*. A backup router participates in a VRRP-E session only when short-path forwarding is enabled.

VRRP-E active-active load-balancing is available in VCS mode and uses flow-hashing techniques to determine the path. All nodes in the VCS are aware of all VRRP-E sessions and the participating RBridges in each session.

Packet routing with short-path forwarding to balance traffic load

When short-path forwarding is enabled, traffic load-balancing is performed because both master and backup devices can be used to forward packets.

FIGURE 90 Short-path forwarding



If you enable short-path forwarding in both master and backup VRRP-E devices, packets sent by Host Server 1 (in the figure) and destined for the Internet cloud through the device on which a VRRP-E backup interface exists can be routed directly to the VRRP-E backup device (blue dotted line) instead of being switched to the master router and then back (red dotted-dash line).

In the figure, load-balancing is achieved using short-path forwarding by dynamically moving the virtual servers between Host Server 1 and Host Server 2.

Short-path forwarding with revert priority

Revert priority is used to dynamically enable or disable VRRP-E short-path forwarding.

If short-path forwarding is configured with revert priority on a backup router, the revert priority represents a threshold for the current priority of the VRRP-E session. When the backup device priority is higher than the configured revert priority, the backup router is able to perform short-path forwarding. If the backup priority is lower than the revert priority, short-path forwarding is disabled.

Configuring VRRP-Ev3 load-balancing

VRRP-Ev3 traffic can be load-balanced using short-path forwarding on the backup devices.

Before configuring VRRP-Ev3 load-balancing, VRRP-Ev3 must be configured on all devices in the VRRP-Ev3 session.

Perform this task on all backup VRRP-Ev3 Layer 3 devices to allow load sharing within an IPv6 VRRP extended group.

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

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an associated RBridge ID to enter RBridge configuration mode for a specific ID.

   ```
   device(config)# rbridge-id 122
   ```

3. Globally enable VRRP-Ev3.

   ```
   device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
   device(config)# ipv6 protocol vrrp-extended
   ```

4. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config-rbridge-id-122)# interface ve 2019
   device(config)# interface ve 2019
   ```

   In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned with a VLAN number of 2019.

5. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2019)# ipv6 address 2001:2019:8192::122/64
   ```

6. Enter the **ipv6 vrrp-extended-group** command with a number to assign a VRRP-E group to the device.

   ```
   device(config-ve-2018)# ipv6 vrrp-extended-group 19
   ```

   In this example, VRRP-Ev3 group configuration mode is entered.

7. Enter the **short-path-forwarding** command with a **revert-priority** value to configure the backup VRRP-E as an alternate path with a specified priority.

   ```
   device(config-vrrp-extended-group-19)# short-path-forwarding revert-priority 50
   ```

   When the backup device priority is higher than the configured **revert-priority** value, the backup router is able to perform short-path forwarding. If the backup priority is lower than the revert priority, short-path forwarding is disabled.

In the following example, short-path forwarding is configured on a backup VRRP-Ev3 device and a revert priority threshold is configured. If the backup device priority falls below this threshold, short-path forwarding is disabled.

```
device# configure
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
device(config-rbridge-id-122)# interface ve 2019
device(config-ve-2019)# ipv6 address 2001:2019:8192::122/64
device(config-ve-2019)# ipv6 vrrp-extended-group 19
device(config-vrrp-extended-group-19)# short-path-forwarding revert-priority 50

device# configure
device(config)# ipv6 protocol vrrp-extended
device(config)# interface ve 2019
device(config-ve-2019)# ipv6 address 2001:2019:8192::122/64
device(config-ve-2019)# ipv6 vrrp-extended-group 19
device(config-vrrp-extended-group-19)# short-path-forwarding revert-priority 50
```

VRRP-Ev3 sub-second failover

VRRP-Ev3 introduces a scale time factor to the advertisement interval that results in sub-second failover times.

In VRRP version 2, an advertisement interval can be set to decrease the time period between advertisements to allow for shorter or longer convergence times. In VRRPv3, a new CLI command is introduced to allow scaling of the advertisement interval timer. When a scaling value is configured, the existing advertisement interval timer value is divided by the scaling value. For example, if the advertisement interval is set to 1 second and the scaling value is set to 10, the new advertisement interval is 100 milliseconds. Using the timer scaling, VRRP-Ev3 sub-second convergence is possible if a master fails.

For each VRRP-Ev3 session, the same advertisement interval and scale value should be used. There are some limits on the number of VRRP sessions configured with advertisement intervals of one second or less, for details see the VRRPv3 Performance and Scalability Metrics section.

> NOTE
> MLX Series devices only support a scaling factor of 10. For interoperability with these devices, use an advertisement interval scale factor of 10.

Configuring sub-second failover using VRRP-Ev3

Configuring a scale factor making the interval between VRRP advertisements to be set in milliseconds allows a sub-second convergence time if a master VRRP device fails.

The configuring sub-second failover using VRRP-Ev3 task is only supported by VRRP-Ev3.

> NOTE
> Increased timing sensitivity as a result of this configuration could cause protocol flapping during periods of network congestion.

1. Enter the **configure** command to access global configuration mode.

   ```
   device# configure
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. To globally enable VRRP-Ev3, enter the **ipv6 protocol vrrp-extended** command.

   ```
   device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
   ```

4. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config-rbridge-id-122)# interface ve 2019
   ```

   In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned with a VLAN number of 2019.

5. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2019)# ipv6 address 2001:2019:8192::122/64
   ```

6. Enter the **ipv6 vrrp-extended-group** command with a number to assign a VRRP-E group to the device.

   ```
   device(config-ve-2018)# ipv6 vrrp-extended-group 19
   ```

   In this example, VRRP-Ev3 group configuration mode is entered.

7. Enter the **advertisement-interval** command with a value to set the time period in seconds between VRRP advertisements.

```
device(config-vrrp-extended-group-19)# advertisement-interval 1
```

8. Enter the **advertisement-interval-scale** command with a value of 1, 2, 5, or 10. The VRRP advertisement interval is divided by this number to set the time period in milliseconds between VRRP advertisements.

```
device(config-vrrp-extended-group-19)# advertisement-interval-scale 10
```

   In this example, the scale number of 10 divided into the advertisement interval of 1 sets the interval between advertisements to 100 milliseconds. If a master VRRP-E device fails, the convergence time to a backup VRRP-E device may be in less than half a second.

The following example demonstrates how to configure a VRRP advertisement interval of 100 milliseconds for an IPv6 VRRP-Ev3 group.

```
device# configure
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
device(config-rbridge-id-122)# interface ve 2019
device(config-ve-2019)# ipv6 address 2001:2019:8192::122/64
device(config-ve-2019)# ipv6 vrrp-extended-group 19
device(config-vrrp-extended-group-19)# advertisement-interval 1
device(config-vrrp-extended-group-19)# advertisement-interval-scale 10
```

Displaying and clearing VRRP-Ev3 statistics

Several show commands can display statistical information about IPv6 VRRP-Ev3 configurations. To reset the IPv6 VRRP-Ev3 statistics, there is a CLI command.

Before displaying statistics, VRRP-Ev3 must be configured and enabled in your network to generate traffic.

Use one or more of the following commands to display VRRP-Ev3 information. The commands do not have to be entered in this order.

1. Use the **exit** command to return to privileged EXEC mode, if required.

2. Enter the **show ipv6 vrrp-extended** command.

```
device# show ipv6 vrrp-extended

============Rbridge-id:122============

Total number of VRRP session(s)   : 1

VRID 19
  Interface: Ve 2019;  Ifindex: 1207961571
  Mode: VRRPE
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 3
  Authentication type: No Authentication
  State: Backup
  Session Master IP Address: fe80::205:33ff:fe79:fb1e
  Virtual IP(s): 2001:2019:8192::1
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: DISABLE  (default: DISABLED)
  Advertise-backup: ENABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Enabled
  Revert Priority: unset; SPF reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Trackport:
    Port(s)                   Priority  Port Status
    =======                   ========  ===========
  Statistics:
    Advertisements: Rx: 102992, Tx: 1716
    Neighbor Advertisements: Tx: 0
```

This example output shows that one IPv6 VRRP-E session is configured.

3. Enter the **show ipv6 vrrp summary** command.

```
device# show ipv6 vrrp summary

============Rbridge-id:122============

Total number of VRRP session(s)   : 2
Master session count  : 1
Backup session count  : 1
Init session count    : 0

VRID  Session  Interface  Admin    Current   State   Short-path  Revert    SPF
                         State    Priority          Forwarding  Priority  Reverted
====  =======  =========  =====    ========  =====   ==========  ========  ========
18    VRRPE    Ve 2018    Enabled  254       Master  Enabled     unset     No
19    VRRPE    Ve 2019    Enabled  100       Backup  Enabled     unset     No

device# show ipv6 vrrp summary

Total number of VRRP session(s)   : 2
Master session count  : 1
Backup session count  : 1
Init session count    : 0

VRID  Session  Interface  Admin    Current   State   Short-path  Revert    SPF
                         State    Priority          Forwarding  Priority  Reverted
====  =======  =========  =====    ========  =====   ==========  ========  ========
18    VRRPE    Ve 2018    Enabled  254       Master  Enabled     unset     No
19    VRRPE    Ve 2019    Enabled  100       Backup  Enabled     unset     No
```

This example shows summary output for the two IPv6 VRRP-E sessions that are configured for virtual routers 18 and 19.

4. Enter the **show ipv6 vrrp 19 detail** command.

```
device# show ipv6 vrrp 19 detail

============Rbridge-id:122============

Total number of VRRP session(s)   : 1

VRID 19
  Interface: Ve 2019;  Ifindex: 1207961571
  Mode: VRRPE
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 3
  Authentication type: No Authentication
  State: Backup
  Session Master IP Address: fe80::205:33ff:fe79:fb1e
  Virtual IP(s): 2001:2019:8192::1
  Virtual MAC Address: 02e0.5200.2513
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: DISABLE  (default: DISABLED)
  Advertise-backup: ENABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Enabled
  Revert-Priority: unset; SPF Reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 103259, Tx: 1721
    Neighbor Advertisements             : Tx: 0
    Session becoming master    : 0
    Advts with wrong interval  : 0
    Prio Zero pkts             : Rx: 0, Tx: 0
    Invalid Pkts Rvcd          : 0
    Bad Virtual-IP Pkts        : 0
    Invalid Authenticaton type : 0
    Invalid TTL Value          : 0
    Invalid Packet Length      : 0
    VRRPE backup advt sent     : 1721
    VRRPE backup advt recvd    : 0

device# show ipv6 vrrp 19 detail

Total number of VRRP session(s)   : 1

VRID 19
  Interface: Ve 2019;  Ifindex: 1207961571
  Mode: VRRPE
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 3
  Authentication type: No Authentication
  State: Backup
  Session Master IP Address: fe80::205:33ff:fe79:fb1e
  Virtual IP(s): 2001:2019:8192::1
  Virtual MAC Address: 02e0.5200.2513
  Configured Priority: unset (default: 100); Current Priority: 100
```

```
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: DISABLE  (default: DISABLED)
  Advertise-backup: ENABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Enabled
  Revert-Priority: unset; SPF Reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 103259, Tx: 1721
    Neighbor Advertisements             : Tx: 0
    Session becoming master    : 0
    Advts with wrong interval  : 0
    Prio Zero pkts             : Rx: 0, Tx: 0
    Invalid Pkts Rvcd          : 0
    Bad Virtual-IP Pkts        : 0
    Invalid Authenticaton type : 0
    Invalid TTL Value          : 0
    Invalid Packet Length      : 0
    VRRPE backup advt sent     : 1721
    VRRPE backup advt recvd    : 0
```

This example shows detailed output for the IPv6 VRRP-E session for virtual router 19.

5. Enter the **show ipv6 vrrp-extended brief** command to display VRRP-Ev3 summary information.

```
device(config)# show ipv6 vrrp-extended brief

Total number of VRRP routers defined: 1
Flags Codes - P:Preempt 2:V2 3:V3 S:Short-Path-Fwd
Intf    VRID CurrPrio Flags State  Master-IPv6 Backup-IPv6 Virtual-IPv6
Address                                        Address     Address
-------------------------------------------------------------------------
1/3     2    100      P3-   Master Local       fd2b::2     fd2b::99
```

6. Enter the **show ipv6 vrrp-extended vrid 1** command to display detailed IPv6 VRRP-E configuration information about VRID 1.

```
device# show ipv6 vrrp-extended vrid 1
Interface 1/1
----------------
auth-type md5-authentication
VRID 1 (index 1)
interface 1/1
state master
administrative-status enabled
mode non-owner(backup)
virtual mac dddd.eeee.ffff (configured)
priority 100
current priority 100
track-priority 5
hello-interval 1 sec
backup hello-interval 60 sec
advertise backup disabled
dead-interval 0 ms
preempt-mode true
virtual ipv6 address 10:20:1::100
```

7. Enter the **clear ipv6 vrrp-extended statistics** command to reset the statistical counters for an IPv6 VRRP-Ev3 session.

```
device# clear ipv6 vrrp-extended statistics
```

8. Enter the **clear ipv6 vrrp statistics** command with the **all** option to reset the statistical counters for all IPv6 VRRP-Ev3 sessions.

```
device# clear ipv6 vrrp statistics all
```

9. Enter the **clear ipv6 vrrp statistics** command with the **session** option to reset the statistical counters for the IPv6 VRRP-Ev3 session for virtual router 19.

```
device# clear ipv6 vrrp statistics session 19
```

VXLAN

- VXLAN Transit Mode....737
- VXLAN Native IP Mode....738

VXLAN Transit Mode

An interface configured in Transit mode filters based on VXLAN inner headers, without stripping the outer headers.

By default, tunneled flows are filtered based on inner headers only when you configure the stripping of outer headers on the ingress interface. Otherwise, filtering is based on the outer headers. In VXLAN Transit mode, Layer 2 and Layer 3 policies are always applied on inner headers, irrespective of whether you have enabled outer header stripping.

If the VXLAN traffic that ingresses on a Transit interface has an inner IPv6 header, the Layer 3 and UDA policies can filter and match on the full 128-bit Source and Destination IP addresses. On regular interfaces, filtering is limited based on lower or upper 64-bits of SIP and DIP.

Non-VXLAN traffic that ingresses on an interface in VXLAN Transit mode is treated as an Ethernet frame with an unknown Layer 3 Protocol. If necessary, you can filter these frames using Layer 2 or UDA policies.

Consider the following:

- Native IP mode and VXLAN Transit mode are mutually exclusive.
- The inner and outer Ethernet header can only be Untagged or C-Tagged. Any other TAG combination fails parsing. Such a frame is classified as an Ethernet frame with an unknown Layer 3 protocol.
- No interface-level editing configurations, except for strip-vxlan, are allowed on an interface in Transit mode.
- VLAN add and delete operations are not allowed on an interface in Transit mode.

You can use the **profile npb-parser** command to select the Default or VXLAN Transit profile.

You can use the **mode vxlan-transit** command to enable VXLAN Transit mode on an interface.

UDA in Transit mode

VXLAN traffic on a Transit interface is parsed differently from VXLAN traffic on a regular interface. The outer headers are not copied into the Token Header Buffer and are not available for policy lookups, including the UDA. The UDA profile on a VXLAN Transit interface does not include any outer headers and is defined to match the inner and payload headers. For example:

For example, here is the UDA profile definition on a regular interface.

```
device(config)# uda-key profile UDA_PROFILE1
device(config-uda-key)# flow header0 ETHERNET header1 IPV4 header2 UDP header3 VXLAN header4
 TUN_ETHERNET header5 IPV4 header6 TCP header7 PAYLOAD4
```

On a Transit interface, the UDA profile is defined as follows:

```
device(config)# uda-key profile UDA_PROFILE2
device(config-uda-key)# flow header0 ETHERNET header1 IPV4 header2 TCP header3 PAYLOAD32
```

In this example, the inner header is IPv6:

```
device(config)# uda-key profile UDA_PROFILE3
device(config-uda-key)# flow header0 ETHERNET header1 IPV6_128 header2 TCP header3 PAYLOAD32
```

VXLAN Native IP Mode

An interface configured in Native (non-tunnel) IP mode parses only the native headers (outer Layer 2, Layer 3, and Layer 4). Parsing stops after the outer Layer 4 header and tunneled headers (if any) are not parsed.

For IPv6 flows, Native IP mode also supports matching and filtering based on full 128-bit Source and Destination addresses. Because tunneled headers are not parsed in Native IP mode, the Token Header Buffer always has sufficient space to copy the full IPv6 SIP and DIP.

Consider the following:

- Native IP mode and VXLAN Transit mode are mutually exclusive.
- The Ethernet header can only be Untagged or C-Tagged. Any other TAG combination fails the parsing. Such a frame is classified as an Ethernet frame with an unknown Layer 3 protocol.
- No interface-level editing configurations, such as strip-vxlan, strip-nvgre, and strip-mpls, apply in Native mode.
- VLAN add and delete operations work in Native mode in the same way that they work in normal mode.

You can use the **mode native-ip** command to enable Native mode on an interface.

UDA in Native IP mode

UDA behaves the same way in Native IP mode as it does on a regular interface, except when you are trying to match an IPv6 flow. Use the header type **IPV6_128** to match an IPv6 flow. For example:

```
device(config)# uda-key profile UDA_PROFILE1
device(config-uda-key)# flow header0 ETHERNET header1 IPV6_128 header2 TCP header3 PAYLOAD32
```