CONFIGURATION GUIDE

Extreme SLX-OS Layer 3 Routing Configuration Guide, 18r.1.00b

Supporting the ExtremeRouting SLX 9850 and ExtremeSwitching SLX 9540 Devices

9035452-00 Rev AA
March 2020

Copyright © 2019 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, please 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: www.extremenetworks.com/support/policies/software-licensing

Contents

Preface **19**
- Conventions 19
  - Notes, cautions, and warnings 19
  - Text formatting conventions 19
  - Command syntax conventions 20
- Documentation and Training 20
  - Training 20
- Getting Help 20
  - Subscribing to Service Notifications 21
- Providing Feedback to Us 21

About This Document **23**
- Supported hardware and software 23
  - Interface module capabilities 23
- What's new in this document 23

ARP **25**
- ARP and Neighbor Discovery overview 25
- Basic ARP configuration 25
  - Modifying the ARP aging timeout 25
  - Enabling and disabling ARP learning 26
  - Creating static ARP entries 26
  - Creating a VRF static ARP entry 27
  - Proxy ARP 27
  - ARP show and clear commands 28
- Dynamic ARP Inspection (DAI) 28
  - ARP poisoning 28
  - DAI functionality 29
  - DAI implementation 30
  - DAI show and clear commands 32
- ARP Guard 32
  - ARP Guard and DAI 32
  - ARP Guard implementation 33
  - ARP Guard show commands 36
- ARP and ND suppression 36
  - Enabling ARP and ND suppression on a VLAN 37
  - Enabling ARP and ND suppression on a bridge domain 37
  - ARP and ND suppression show and clear commands 38

IP Addressing **39**
- IP addressing overview 39
  - IPv4 address classes 39
  - IP Interfaces 39
  - The ARP cache 40
  - 31-bit subnet masks on point-to-point networks 40
  - IP unnumbered interfaces 40
- Basic IP parameters and defaults 41
  - Parameter changes in effect 41

IP global parameters 41
IP interface parameters 43
Assigning IP addresses to interfaces 44
Assigning an IP address to a loopback 44
Assigning IP addresses to Ethernet interfaces 45
Deleting an IP address from an interface 46
Router-ID IP addresses 46
Assigning a router-ID IP address 46
DNS 47
DNS Server 47
DNS Resolver 47
DNS Gateway Addresses 47
Configuring DNS 47
Source IP address for various packet types 48
IPv4 MTU settings 49
IPv4 MTU and maximum frame size 49
Changing IPv4 MTU on an interface 49
IP addressing Show and Clear commands 50

IPv6 Addressing 51
IPv6 addressing overview 51
Configuring an IPv6 address 52
IPv6 management 52
Configuring IPv6 management ACLs 52
Specifying an IPv6 SNMP trap receiver 53
Secure Shell, SCP, and IPv6 53
IPv6 Telnet 53
Establishing a Telnet session from an IPv6 host 53
IPv6 traceroute 53
Restricting Web management access 54
IPv6 neighbor discovery configuration 54
IPv6 neighbor discovery configuration notes 54
Router advertisement and solicitation messages 54
Neighbor redirect messages 55
Duplicate Address Detection (DAD) 55
Configuring IPv6 static neighbor entries 55
IPv6 router advertisement parameters 55
Setting IPv6 router advertisement parameters 56
Setting flags in IPv6 router advertisement messages 56
Sending or Suppressing IPV6 Router Advertisements and Addresses 57
IPv6 router advertisement preference support 57
Configuring IPv6 RA preference 58
Reachable time for remote IPv6 nodes 58
Configuring reachable time for remote IPv6 nodes 58
Configuring an IPv6 domain name and server 58
IPv6 MTU 59
Changing IPv6 MTU 59
Information about IPv6 prefix list 59
Configuring an IPv6 prefix list 60
Displaying prefix list information 60
Displaying global IPv6 information 60

Displaying IPv6 interface information 61
Displaying IPv6 neighbor information 61
Displaying the IPv6 route table 61
Clearing global IPv6 information 62
Clearing IPv6 neighbor information 62
Clearing IPv6 routes from the IPv6 route table 62

IPv4 Static Routing 63
Overview of static routing 63
Static route states follow port states 64
Configuring a basic IP static route 64
Adding metrics to a static route 65
Configuring a physical interface as next hop 65
Configuring a virtual interface as next hop 66
Configuring a static route with a VRF as next hop 66
Configuring a static route for use with a route map 67
Configuring a null route 67
Configuring a default static route 69
Configuring load sharing and redundancy 69
Displaying IPv4 static routes 71

IPv6 Static Routing 73
Overview of static routing 73
Static route states follow port states 74
Configuring a basic IPv6 static route 74
Removing an IPv6 static route 75
Removing an IPv6 static route in a non-default VRF 76
Configuring an interface as next hop 76
Configuring a virtual interface as next hop 77
Configuring a VRF as next hop for an IPv6 static route 78
Adding metrics to an IPv6 static route 78
Configuring a null route 79
Configuring a default static route 81
Configuring load sharing and redundancy 81
Adding an IPv6 static route tag for use with route-maps 83
Displaying information on IPv6 static routes 84

DHCPv4 85
DHCPv4 overview 85
IP DHCP Relay function 85
Configuring IP DHCP Relay 86
DHCP relay agent supported user scenarios 88
DHCP server local 88
DHCP server remote 88
DHCP servers in a network 88
Multiple DHCP servers 88
DHCP server in different VRFs 89
DHCP relay agent information option 82 89
Enabling the DHCP Relay Agent Information option 90
Configuring DHCP relay gateway address 90
VRF support 91
Supported VRF configuration examples 91

VRF configuration examples to avoid ... 92
Displaying IP DHCP Relay statistics ... 92
Displaying IP DHCP Relay addresses on specific devices ... 93
Displaying IP DHCP relay addresses for an interface ... 93
Clearing IP DHCP relay statistics ... 94

DHCPv6 ... 95
DHCPv6 overview ... 95
DHCP relay agent for IPv6 ... 95
DHCPv6 multicast addresses and UDP ports ... 95
Multicast addresses ... 95
UDP ports ... 96
DHCPv6 address assignment ... 96
Basic DHCPv6 relay assignment ... 96
DHCPv6 prefix delegation ... 96
Relay chaining ... 96
Relay-message option ... 96
Remote-ID option ... 96
Interface-ID option ... 96
DHCPv6 message format ... 96
Configuring IPv6 DHCP relay ... 98
Displaying DHCPv6 relay addresses on specific devices ... 99
Displaying DHCPv6 relay addresses for an interface ... 99
Displaying IPv6 DHCP relay statistics ... 100
Clearing IP DHCPv6 relay statistics ... 100

BFD ... 101
Bidirectional Forwarding Detection overview ... 101
General BFD considerations and limitations ... 102
BFD for Layer 3 protocols ... 102
BFD considerations and limitations for Layer 3 protocols ... 103
BFD for Layer 3 protocols on virtual Ethernet interfaces ... 104
Configuring BFD on an interface ... 104
Disabling BFD on an interface ... 105
BFD for BGP ... 105
BFD for BGP session creation and deletion ... 106
Configuring BFD session parameters for BGP ... 106
Enabling BFD sessions for a specified BGP neighbor ... 107
Enabling BFD sessions for a specified BGP neighbor in a nondefault VRF ... 107
Enabling BFD sessions for a specified BGP peer group ... 108
BFD for OSPF ... 109
BFD for OSPF session creation and deletion ... 109
Enabling BFD on a specified OSPFv2-enabled interface ... 110
Configuring BFD for OSPFv2 globally ... 110
Configuring BFD for OSPFv2 globally in a nondefault VRF instance ... 111
Enabling BFD on a specified OSPFv3-enabled interface ... 111
Configuring BFD for OSPFv3 globally ... 112
Configuring BFD for OSPFv3 globally in a nondefault VRF instance ... 112
BFD for IS-IS ... 113
Enabling BFD on a specified IS-IS-enabled interface ... 113
Configuring BFD for IS-IS globally ... 113

Displaying BFD information....114
BGP4....117
BGP4 overview....118
BGP4 peering....119
BGP4 message types....119
OPEN message....119
UPDATE message....120
NOTIFICATION message....120
KEEPALIVE message....121
REFRESH message....121
BGP4 attributes....121
BGP4 best path selection algorithm....121
Device ID....123
BGP global mode....123
Configuring a local AS number....124
IPv4 unicast address family....124
Neighbor configuration....125
Configuring BGP4 neighbors....126
Peer groups....126
Configuring BGP4 peer groups....127
Advertising the default BGP4 route....127
Four-byte AS numbers....128
Enabling ASN capability globally....128
Enabling ASN capability for a BGP4 neighbor....129
Cooperative BGP4 route filtering....129
BGP4 parameters....130
Route redistribution....131
Redistributing routes into BGP4....131
Advertised networks....131
Importing routes into BGP4....132
Static networks....132
Configuring a static network....132
Route reflection....133
Configuring a cluster ID for a route reflector....133
Configuring a route reflector client....134
Route flap dampening....134
Aggregating routes advertised to BGP neighbors....135
Advertising the default BGP4 route....135
Advertising the default BGP4 route to a specific neighbor....136
Multipath load sharing....136
Specifying the weight added to received routes....137
Using the IPv4 default route as a valid next hop for a BGP4 route....137
Adjusting defaults to improve routing performance....138
Next-hop recursion....138
Enabling next-hop recursion....138
Route filtering....139
BGP regular expression pattern-matching characters....139
Timers....140
Enabling BGP4 in a non-default VRF....140
BGP4 outbound route filtering....141

Configuring BGP4 outbound route filtering....141
Enabling BGP4 cooperative route filtering....142
BGP4 confederations....143
Configuring BGP4 confederations....144
BGP community and extended community....145
Defining BGP4 extended communities....145
Applying a BGP4 extended community filter....146
BGP4 graceful restart....147
Configuring BGP4 graceful restart....148
BGP additional-paths....149
Configuration of BGP additional-paths....151
Advantages of BGP additional-paths....151
Considerations and limitations for BGP additional-paths....152
Upgrade and downgrade considerations for BGP additional-paths....152
Configuring BGP additional-paths and additional-path selection at address-family level....152
Configuring BGP additional-paths and additional-path advertisement at neighbor level....153
Configuring additional-paths and advertisement of additional-paths at peer group level....155
Filtering additional-paths advertised at route-map level....156
Disabling BGP additional-paths capability for a neighbor....157
BGP best-external route....158
Configuring BGP best-external route at address-family level....159
Auto shutdown of BGP neighbors on initial configuration....160
Configuring auto shutdown of BGP neighbors on initial configuration....160
Disabling the BGP4 peer shutdown state....161
Generalized TTL Security Mechanism support....161
Assumptions and limitations....162
Configuring GTSM for BGP4....162
Disabling the BGP AS_PATH check function....163
Using route maps....163
Matching on an AS-path....164
Matching on a community ACL....165
Matching on a destination network....165
Matching on a BGP4 static network....166
Matching on a next-hop device....167
Matching on an interface....167
Using route-map continue statements....167
Route-map continue statement for BGP4 routes....169
Using a route map to configure dampening....169
Clearing diagnostic buffers....169
Displaying BGP4 statistics....170
Displaying BGP4 neighbor statistics....173
Clearing BGP4 dampened paths....175

BGP4+....177

BGP4+ overview....177
BGP global mode....178
IPv6 unicast address family....179
BGP4+ neighbors....180
Configuring BGP4+ neighbors using global IPv6 addresses....180
Configuring BGP4+ neighbors using link-local addresses....181
BGP4+ peer groups....182

Configuring BGP4+ peer groups ... 182
Configuring a peer group with IPv4 and IPv6 peers ... 183
Enabling ASN capability globally ... 184
Enabling ASN capability for a BGP4+ neighbor ... 185
Importing routes into BGP4+ ... 185
Advertising the default BGP4+ route ... 186
Advertising the default BGP4+ route to a specific neighbor ... 186
Using the IPv6 default route as a valid next hop for a BGP4+ route ... 187
BGP4+ next hop recursion ... 188
Enabling next-hop recursion ... 188
BGP4+ NLRIs and next hop attributes ... 189
BGP4+ route reflection ... 189
Configuring a cluster ID for a route reflector ... 189
Configuring a route reflector client ... 190
BGP4+ route aggregation ... 190
Aggregating routes advertised to BGP neighbors ... 191
BGP4+ multipath ... 191
Enabling load-balancing across different paths ... 192
Route maps ... 192
Configuring a route map for BGP4+ prefixes ... 192
Redistributing prefixes into BGP4+ ... 194
Redistributing routes into BGP4+ ... 194
Specifying the weight added to BGP4+ received routes ... 195
Enabling BGP4+ in a non-default VRF ... 195
BGP4+ outbound route filtering ... 196
Configuring BGP4+ outbound route filtering ... 196
BGP4+ confederations ... 197
Configuring BGP4+ confederations ... 198
BGP4+ extended community ... 198
Defining BGP4+ extended communities ... 199
Applying a BGP4+ extended community filter ... 200
BGP4+ graceful restart ... 201
Configuring BGP4+ graceful restart ... 201
Auto shutdown of BGP neighbors on initial configuration ... 203
Configuring auto shutdown of BGP neighbors on initial configuration ... 203
Disabling the BGP4+ peer shutdown state ... 203
Generalized TTL Security Mechanism support ... 204
Assumptions and limitations ... 204
Configuring GTSM for BGP4+ ... 205
Disabling the BGP AS_PATH check function ... 205
Displaying BGP4+ statistics ... 206
Displaying BGP4+ neighbor statistics ... 208
Clearing BGP4+ dampened paths ... 210

BGP EVPN for IP Fabrics ... 213
Overview ... 213
IP Fabrics topologies ... 213
IP Fabrics features ... 215
BGP EVPN use cases ... 215
BGP EVPN control plane ... 216
Supported route types ... 216

Supported BGP functionalities....216
Supported data plane encapsulation types....216
Supported service-interface model....217
BGP EVPN configuration examples....217
BGP EVPN neighbor examples....217
BGP EVPN instance configuration....220
BGP routing tables....222
BGP EVPN-based MCT cluster formation....223
Dependency on the EVPN instance....223
BGP EVPN peering for the MCT cluster....224
Handling of ES/AD routes in BGP....225
BGP EVPN-based VXLAN overlay....225
Underlay architectures....226
VXLAN overlay configuration....227
Dynamic VTEP discovery....228
MCT and logical VTEP....229
EVPN next-hop resolution and tunnel-EVI membership....230
Layer 2 (MAC) route exchange....231
MAC move detection and dampening....231
Automatic restoration of dampened routes....232
Aliasing in MPLS....232
Conversational MAC learning....234
Example output for "show mac-address-table"....234
ARP/ND (MACIP) route exchange....235
Conversational ARP learning....235
EVPN prefix route exchange and Multi-VRF support....236
Symmetric or asymmetric routing....237
Import/export route-map filtering....237
Conversion of MACIP to host route to avoid traffic tromboning....237
Avoiding traffic tromboning in an MCT cluster....238
BGP EVPN VXLAN data center interconnect....239
Layer 2 and Layer 3 control-plane extension....240
Layer 2 handoff....240
Layer 3 handoff....240
Static anycast gateway....241
IP unnumbered interface....243
High availability....244

Generic Routing Encapsulation....245
GRE tunnels....245
Limitations and restrictions....246
Configuring a Tunnel....246
Binding a tunnel to a VE interface....247
Enabling tunnel statistics....248
Configuring GRE tunnel keepalive....248
Configuring differentiated services codepoint....249
Configuring MTU....249
Displaying tunnel statistics....250
Clearing tunnel statistics....251

IS-IS (IPv4)....253

IS-IS overview 254
Relationship to the IP route table 254
Intermediate systems and end systems 255
Domain and areas 255
Level-1 routing and Level-2 routing 256
Neighbors and adjacencies 256
Designated IS 256
Broadcast pseudonode 257
Route calculation and selection 257
Three-way handshake for point-to-point adjacencies 257
IS-IS CLI levels 258
Enabling IS-IS globally 258
Configuring the IS-IS IPv4 unicast address family 259
Overload bit 259
Setting the overload bit 260
Authentication 260
Configuring authentication 261
Changing the IS-IS level globally 261
Logging adjacency changes 262
Complete Sequence Numbers PDU interval 262
Configuring the CSNP interval 262
Changing the maximum LSP lifetime 263
Changing the LSP refresh interval 263
Changing the LSP generation interval 264
Changing the LSP interval and retransmit interval 264
Disabling IS-IS name mapping capability 265
Logging invalid LSP packets received 265
Changing the SPF timer 265
Configuring the IS-IS flooding mechanism 266
Configuring IS-IS PSPF exponential back-off 267
Hello padding 267
Disabling hello padding globally 268
Partial SPF optimizations 268
Disabling partial SPF optimizations 268
Incremental SPF optimizations 269
Disabling incremental shortcut SPF optimizations 269
IS-IS incremental shortcut LSP SPF optimization 269
Disabling incremental SPF optimizations 270
Maximum number of load sharing paths 270
Changing the maximum number of load sharing paths 270
Default route advertisement 271
Enabling advertisement of a default route 271
Using a route map to advertise a default route 272
IS-IS administrative distance 272
Changing the administrative distance for IPv4 IS-IS 273
Configuring summary addresses 273
IPv4 IS-IS route redistribution 274
Redistributing routes into IPv4 IS-IS 274
Default redistribution metric 275
Changing the default redistribution metric 275

Configuring the default link metric value globally....276
IS-IS metric styles....276
Changing the metric style....277
IS-IS non-stop routing....277
Enabling non-stop routing....277
Limitations of IS-IS non-stop routing....278
DIS hello interval....278
Enabling IS-IS for an Interface....279
Configuring authentication on an IS-IS interface....279
Disabling hello padding for an IS-IS interface....280
Changing the IS-IS level on an IS-IS interface....280
Changing the hello multiplier for an IS-IS interface....281
Changing the hello interval for an IS-IS interface....281
IS-IS point-to-point over Ethernet....282
Enabling IS-IS point-to-point over Ethernet....282
Displaying IS-IS statistics....284

IS-IS (IPv6)....287
IPv6 IS-IS single-topology mode....288
IS-IS CLI levels....289
Enabling IPv6 IS-IS globally....289
Configuring the IS-IS IPv6 unicast address family....290
Configuring IPv6 IS-IS single topology....290
Setting the overload bit....291
Configuring authentication....291
Changing the IS-IS level globally....292
Logging adjacency changes....292
Configuring the CSNP interval....293
Changing the maximum LSP lifetime....293
Changing the LSP refresh interval....293
Changing the LSP generation interval....294
Changing the LSP interval and retransmit interval....294
Disabling IS-IS name mapping capability....295
Logging invalid LSP packets received....295
Changing the SPF timer....296
Configuring the IS-IS flooding mechanism....296
Configuring IS-IS PSPF exponential back-off....297
Disabling hello padding globally....297
Disabling partial SPF optimizations....298
Disabling incremental SPF optimizations....298
Disabling incremental shortcut SPF optimizations....298
Maximum number of load sharing paths....299
Changing the maximum number of load sharing paths....299
Default route advertisement....300
Enabling advertisement of a default route....300
Using a route map to advertise a default route....300
IPv6 IS-IS administrative distance....301
Changing the administrative distance for IPv6 IS-IS....301
Configuring summary prefixes....302
Redistributing routes into IPv6 IS-IS....302
Redistributing routes into IPv6 IS-IS....303

Default redistribution metric ... 304
Changing the default redistribution metric ... 304
Configuring the default link metric value globally ... 305
IPv6 metric behavior with multi-topology configuration ... 305
IS-IS metric styles ... 305
IPv6 IS-IS non-stop routing ... 306
Enabling non-stop routing ... 306
Limitations of IPv6 IS-IS non-stop routing ... 306
IPv6 protocol-support consistency checks ... 307
Enabling IS-IS and assigning an IPv6 address to an interface ... 307
Configuring authentication on an IS-IS interface ... 308
Disabling hello padding for an IS-IS interface ... 308
Changing the IS-IS level on an IS-IS interface ... 309
Changing the hello multiplier for an IS-IS interface ... 309
Changing the hello interval for an IS-IS interface ... 310
Changing the metric added to advertised routes for an IS-IS interface ... 310
Disabling IPv6 protocol-support consistency checks ... 311
IPv6 IS-IS Multi-Topology ... 311
Configuration considerations for IPv6 IS-IS MT ... 312
Migrating to IPv6 IS-IS MT ... 312
Maintaining MT adjacencies ... 312
Forming adjacencies on the point-to-point interfaces ... 313
Forming adjacencies on the broadcast interfaces ... 313
New TLV attributes ... 313
Enabling IPv6 IS-IS MT ... 313
Configuration example to deploy IPv6 IS-IS MT ... 313
Configuration commands to enable IPv6 IS-IS MT on device D1 ... 314
Configuration commands to enable IPv6 IS-IS MT on device D2 ... 314
Configuration commands to enable IPv6 IS-IS MT on device E2 ... 314
Configuration commands to enable IPv6 IS-IS MT on device C2 ... 315
Displaying IS-IS statistics ... 316

OSPFv2 ... 319
OSPFv2 overview ... 319
Autonomous System ... 320
OSPFv2 components and roles ... 320
Area Border Routers ... 320
Autonomous System Boundary Routers ... 321
Designated routers ... 321
Enabling OSPFv2 ... 322
Backbone area ... 322
Assigning OSPFv2 areas ... 323
Area range ... 323
Assigning an area range ... 323
Area types ... 324
Stub area and totally stubby area ... 324
Disabling summary LSAs for a stub area ... 325
Not-so-stubby area (NSSA) ... 325
Configuring an NSSA ... 326
Configuring a summary-address for the NSSA ... 327
Assigning interfaces to an area ... 327

Link state advertisements....328
Configuring an MD5 password and authentication change hold time for an OSPFv2 interface....328
Virtual links....328
Configuring virtual links....329
Default route origination....330
External route summarization....331
Modifying Shortest Path First timers....331
OSPFv2 administrative distance....332
OSPFv2 LSA refreshes....332
Configuring the OSPFv2 LSA pacing interval....333
OSPFv2 graceful restart....333
Disabling OSPFv2 graceful restart....334
Re-enabling OSPFv2 graceful restart....334
Disabling OSPFv2 graceful restart helper....335
OSPFv2 non-stop routing....335
Enabling OSPFv2 NSR....335
Redistributing routes into OSPFv2....336
OSPFv2 type 3 LSA filtering....336
Usage and configuration guidelines....337
Configuring OSPFv2 type 3 LSA filtering....338
OSPFv2 over VRF....339
Enabling OSPFv2 in a non-default VRF....339
Configuring the OSPFv2 Max-Metric Router LSA....339
Re-enabling OSPFv2 compatibility with RFC 1583....340
Changing default settings....340
Disabling and re-enabling OSPFv2 event logging....340
Understanding the effects of disabling OSPFv2....341
Disabling OSPFv2....341
Displaying OSPFv2 results....342
Supported scale for OSPFv2 and performance considerations....345

OSPFv3....347

OSPFv3 overview....347
Configuring the router ID....348
Enabling OSPFv3....348
Configuring OSPFv3....348
OSPFv3 areas....349
Backbone area....349
Area range....349
Area types....349
Assigning OSPFv3 areas....350
Assigning OSPFv3 areas to interfaces....350
Stub area and totally stubby area....351
Configuring a stub area....352
Not-so-stubby area....352
Configuring an NSSA....353
LSA types for OSPFv3....353
Virtual links....354
Virtual link source address assignment....356
Configuring virtual links....356
OSPFv3 route redistribution....357

Redistributing routes into OSPFv3 ... 358
Default route origination ... 359
Configuring default external routes ... 359
Disabling and re-enabling OSPFv3 event logging ... 359
Filtering OSPFv3 routes ... 360
SPF timers ... 360
Modifying SPF timers ... 360
OSPFv3 administrative distance ... 361
Configuring administrative distance based on route type ... 361
Changing the reference bandwidth for the cost on OSPFv3 interfaces ... 362
OSPFv3 LSA refreshes ... 363
Configuring the OSPFv3 LSA pacing interval ... 363
External route summarization ... 363
OSPFv3 over VRF ... 364
Enabling OSPFv3 in a non-default VRF ... 364
Setting all OSPFv3 interfaces to the passive state ... 365
OSPFv3 graceful restart helper ... 365
Disabling OSPFv3 graceful restart helper ... 365
Re-enabling OSPFv3 graceful restart helper ... 366
OSPFv3 non-stop routing ... 366
Enabling OSPFv3 NSR ... 366
OSPFv3 max-metric router LSA ... 367
Configuring the OSPFv3 max-metric router LSA ... 367
IPsec for OSPFv3 ... 368
IPsec for OSPFv3 configuration ... 369
Configuring IPsec on an OSPFv3 area ... 369
Configuring IPsec on an OSPFv3 interface ... 370
Configuring IPsec on OSPFv3 virtual links ... 371
Specifying the key rollover and key add-remove timers ... 371
Displaying OSPFv3 results ... 372
Supported scale for OSPFv3 and performance considerations ... 377

Multi-VRF ... 379
Multi-VRF overview ... 379
Configuring Multi-VRF ... 381
Configuring a VRF instance ... 381
Starting a routing process for a VRF ... 381
Assigning a Layer 3 interface to a VRF ... 382
Assigning a loopback interface to a VRF ... 382
Verifying a Multi-VRF configuration ... 383
Removing a VRF configuration ... 383
Configuring the maximum number of routes ... 384
Multi-VRF configuration example ... 384
Multi-VRF with eBGP and OSPF: Configuring PE1 ... 386
Multi-VRF with eBGP and OSPF: Configuring PE2 ... 389
Multi-VRF with eBGP and OSPF: Configuring CE1 and CE2 ... 390
Multi-VRF with eBGP and OSPF: Configuring CE3 and CE4 ... 390
Inter-VRF route leaking ... 390
Dynamic route-leak restrictions ... 391
Inter-VRF route conflicts ... 391
Displaying inter-VRF route leaking ... 392

Configuring static inter-VRF route leaking....393
Configuring dynamic inter-VRF route leaking....396
Internet route scaling overview....399
Configuring profile route....399
VRRPv2....401
VRRPv2 overview....401
VRRP terminology....403
SLX-OS VRRP virtual router MAC address limitations....404
SLX-OS VRRP and VRRP-E interoperability....404
VRRP hold timer....405
VRRP interval timers....405
VRRP authentication....405
ARP and VRRP control packets....406
Enabling a master VRRP device....406
Enabling a backup VRRP device....408
VRRP multigroup clusters....408
Configuring multigroup VRRP routing....409
Tracked ports and track priority with VRRP and VRRP-E....411
Configuring VRRP port tracking....412
VRRP backup preemption....413
Enabling VRRP backup preemption....413
Accept mode for backup VRRP devices....414
Disabling accept mode on a backup VRRP device....414
Virtual router MAC address....414
Configuring unique virtual MAC addresses per VRID....415
VRRP-Ev2 overview....416
Enabling a VRRP-E device....417
Configuring MD5 authentication on IPv4 VRRP-E interfaces....418
Track routes and track priority with VRRP-E....419
Configuring VRRP-E route tracking....420
VRRP-E load-balancing using short-path forwarding....421
Packet routing with short-path forwarding to balance traffic load....421
Short-path forwarding with revert priority....422
Configuring VRRP-E load-balancing using short-path forwarding....423
Displaying VRRPv2 information....423
Clearing VRRPv2 statistics....425
VRRPv3....427
VRRPv3 overview....427
Enabling IPv6 VRRPv3....428
Enabling IPv4 VRRPv3....429
Tracked ports and track priority with VRRP and VRRP-E....430
Port tracking using IPv6 VRRPv3....430
VRRP hold timer....431
Configuring VRRP hold timer support....432
Accept mode for backup VRRP devices....433
Disabling accept mode on a backup VRRP device....433
Alternate VRRPv2 checksum for VRRPv3 IPv4 sessions....434
Enabling the v2 checksum computation method in a VRRPv3 IPv4 session....434
VRRPv3 router advertisement suppression....435

Disabling VRRPv3 router advertisements....435
Displaying VRRPv3 statistics....435
Clearing VRRPv3 statistics....437
VRRP-Ev3 Overview....438
Enabling IPv6 VRRP-Ev3....438
Configuring MD5 authentication on IPv6 VRRP-Ev3 interfaces....439
VRRP-E load-balancing using short-path forwarding....440
Packet routing with short-path forwarding to balance traffic load....440
Short-path forwarding with revert priority....441
Configuring VRRP-Ev3 load-balancing....442
Displaying and clearing VRRP-Ev3 statistics....442

VXLAN Layer 3 Gateway....445
Overview....445
VXLAN Layer 3 gateway modes....447
Configuring VXLAN Layer 3 gateway....452
Configuring TCAM profiles to support Layer 3 gateway....452
Configuring a single-VTEP static VLAN/VE Layer 3 gateway....452
Configuring a single-VTEP static BD/VE Layer 3 gateway....453
Configuring an EVPN Layer 3 gateway for MAC IP routes....454
Configuring an EVPN Layer 3 VNI....455
Configuring an EVPN LVTEP for MAC IP routes....456
Configuring an EVPN Layer 3 VNI for LVTEP....457
Example show and clear commands for VXLAN Layer 3 gateway....459
BD VE support for VXLAN Layer 3 gateway....463
BD VE overview....463
Configuring BD VE support for VXLAN Layer 3 gateway....468

QoS for VXLAN Layer 2 and Layer 3 gateway interconnections....475
Configuring QoS for VXLAN Layer 2 and Layer 3 gateway interconnections....476

QoS for VXLAN Layer 3 gateways....477
Configuring QoS for VXLAN Layer 3 gateways....478

Preface

- Conventions 19
- Documentation and Training 20
- Getting Help 20
- Providing Feedback to Us 21

This section discusses the conventions used in this guide, ways to provide feedback, additional help, and other Extreme Networks® publications.

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

To find Extreme Networks product guides, visit our documentation pages at:

Current Product Documentation	www.extremenetworks.com/documentation/
Archived Documentation (for earlier versions and legacy products)	www.extremenetworks.com/support/documentation-archives/
Release Notes	www.extremenetworks.com/support/release-notes
Hardware/Software Compatibility Matrices	https://www.extremenetworks.com/support/compatibility-matrices/
White papers, data sheets, case studies, and other product resources	https://www.extremenetworks.com/resources/

Training

Extreme Networks offers product training courses, both online and in person, as well as specialized certifications. For more information, 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: 1-800-998-2408 (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 and/or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any action(s) 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

Subscribing 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 with your information (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. Click **Submit**.

Providing Feedback to Us

Quality is our first concern at Extreme Networks, and we have 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 especially want to know about:

- Content errors or confusing or conflicting information.
- Ideas for improvements to our documentation so you can find the information you need faster.
- Broken links or usability issues.

If you would like to provide feedback to the Extreme Networks Information Development team, you can do so in two ways:

- Use our short online feedback form at https://www.extremenetworks.com/documentation-feedback/.
- Email us at documentation@extremenetworks.com.

Please 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....23
- What's new in this document....23

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 this release, documenting all possible configurations and scenarios is beyond the scope of this document.

The following hardware platforms are supported by this release:

- ExtremeRouting SLX 9850-4 router
- ExtremeRouting SLX 9850-8 router
- ExtremeSwitching SLX 9540 switch

To obtain information about other releases, refer to the documentation specific to that release.

Interface module capabilities

The following table lists the supported capabilities for the following SLX 9850 interface modules:

- BR-SLX9850-10Gx72S-M
- BR-SLX9850-100Gx36CQ-M
- BR-SLX9850-10Gx72S-D
- BR-SLX9850-100Gx36CQ-D
- BR-SLX9850-100Gx12CQ-M

TABLE 1 SLX 9850 interface modules capabilities

Capability	Modular interface module
MPLS	Yes
Packet buffer memory per interface module	12GB (BR-SLX9850-10Gx72S-M) 36GB (BR-SLX9850-100Gx36CQ-M) 8GB (BR-SLX9850-10Gx72S-D) 24GB (BR-SLX9850-100Gx36CQ-D) 8GB (BR-SLX9850-100Gx12CQ-M)

What's new in this document

The following table includes description of information changed to this guide for the Extreme SLX-OS 18r.1.00b software release.

TABLE 2 Summary of changes in the SLX-OS 18r.1.00b release

Change	Description	Described in
IPv6 MTU	The **ipv6 mtu** command has been deprecated. The **ip mtu** command applies for both IPv4 and IPv6 MTU.	IPv6 MTU on page 59

The following table includes descriptions of new information added to this guide for the Extreme SLX-OS 18r.1.00 software release.

TABLE 3 Summary of enhancements in the SLX-OS 18r.1.00 release

Feature	Description	Described in
BGP EVPN for IP Fabrics	Various feature enhancements	BGP EVPN for IP Fabrics on page 213

For complete information, refer to the *Release Notes*.

ARP

- ARP and Neighbor Discovery overview....25
- Basic ARP configuration....25
- Dynamic ARP Inspection (DAI)....28
- ARP Guard....32
- ARP and ND suppression....36

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
```

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

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

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 **interface ve** command to access VE configuration mode.

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

3. 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.

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.

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
```

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
```

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
```

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

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

ARP show and clear commands

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

TABLE 4 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 5 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 32

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

TABLE 6 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

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

TABLE 7 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.
- VLANs supported for DAI include:
  - 802.1Q VLANs
  - VE interfaces under virtual routing and forwarding (VRF). Both default and non-default VRFs are supported.
- DAI is not supported for management interfaces.
- 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 MLAGs) (in switchport mode)

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 ethernet 2/3
```

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

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

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

```
device(conf-if-eth-2/3)# 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 **vlan** command to access configuration mode on the VLAN for which you are enabling DAI.

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

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

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

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

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

DAI show and clear commands

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

TABLE 8 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.
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 9 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 28.

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 10 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 11 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

TABLE 11 Profile Support for ARP Guard (continued)

Profile	Basic ARP Guard	ARP Guard with Rate Limiting	ARP Guard on VPLS Endpoint	ARP Guard on VPLS End-point with Rate Limiting
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 12 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 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—disabled by default—on the respective VLANs or BDs.

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 **vlan** command to access VLAN configuration mode.

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

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

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

   To disable ARP suppression, enter **no suppress-arp**.
4. 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)# 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 13 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 14 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.

IP Addressing

- IP addressing overview....39
- Basic IP parameters and defaults....41
- Assigning IP addresses to interfaces....44
- Router-ID IP addresses....46
- DNS....47
- Source IP address for various packet types....48
- IPv4 MTU settings....49
- IP addressing Show and Clear commands....50

IP addressing overview

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.

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.

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 on page 243.

Basic IP parameters and defaults

The following protocols are disabled by default:

- Route exchange protocols (OSPF, IS-IS, BGP4)
- Multicast protocols (IGMP, PIM-SM)
- 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 15 IP global parameters

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.	25 minutes

TABLE 15 IP global parameters (continued)

Parameter	Description	Default
	Note: You also can change the ARP age on an individual interface basis. Refer to IP interface parameters on page 43.	
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
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 43.	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 43.	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,	Enabled

TABLE 15 IP global parameters (continued)

Parameter	Description	Default
	destination IP address, source IP address, and IP protocol. **Note:** Load sharing is sometimes called Equal Cost Multi Path (ECMP).	
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/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.	The lowest-numbered IP address on the 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 16 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 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 41.	25 minutes
Directed broadcast forwarding	Locally overrides the global setting. Refer to IP global parameters on page 41.	Disabled

TABLE 16 IP interface parameters (continued)

Parameter	Description	Default
ICMP Redirect messages	Locally overrides the global setting. Refer to IP global parameters on page 41.	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 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

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.

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
```

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-list mycompany.com
```

3. 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
```

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

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

5. (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 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.

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 1/5
```

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

```
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 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
```

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 17 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 18 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

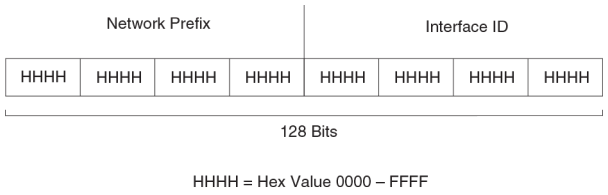
- IPv6 addressing overview....51
- Configuring an IPv6 address....52
- IPv6 management....52
- IPv6 neighbor discovery configuration....54
- Configuring an IPv6 domain name and server....58
- IPv6 MTU....59
- Information about IPv6 prefix list....59
- Displaying global IPv6 information....60
- Clearing global IPv6 information....62

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 0/1
```

3. Configure a global IPv6 address for the interface.

```
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-eth-0/1)# ipv6 address FE80::240:D0FF:FE48:4672 link-local
```

5. Return to Privileged EXEC mode.

```
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)# 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.

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-0S 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.

- Neighbor discovery is not supported on tunnel interfaces.

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.

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 ethernet 3/5.

```
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 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
```

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 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.

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
```

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

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 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.

Displaying prefix list information

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

```
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 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).

IPv4 Static Routing

- Overview of static routing....63
- Configuring a basic IP static route....64
- Adding metrics to a static route....65
- Configuring a physical interface as next hop....65
- Configuring a virtual interface as next hop....66
- Configuring a static route with a VRF as next hop....66
- Configuring a static route for use with a route map....67
- Configuring a null route....67
- Configuring a default static route....69
- Configuring load sharing and redundancy....69
- Displaying IPv4 static routes....71

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, OSPF, BGP4, and ISIS 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 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.

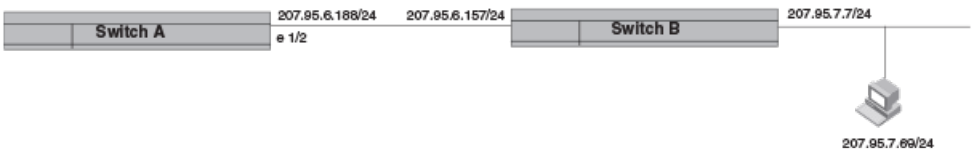
Static route states follow port states

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



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.

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 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.

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
```

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. 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 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
```

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
```

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
```

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
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 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.

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
```

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 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.

```
device# configure terminal
device(config)# 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 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
   ```

   This example routes traffic for IP address 10.0.0.0/24 through VRF red over port 1/1.

   ```
   device(config)# 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
   ```

   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.

```
device(config)# ip route 10.0.0.0/24 next-hop-vrf red ethernet 2/1
```

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 **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
   ```

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.

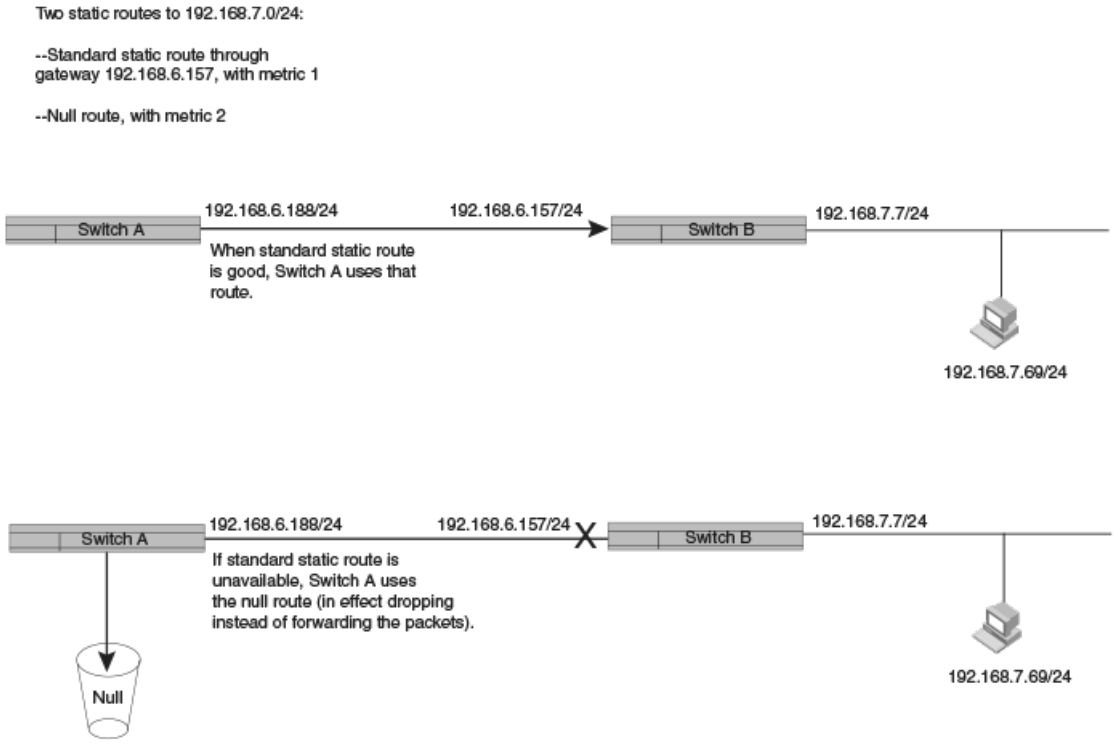
```
device# configure terminal
device(config)# ip route 10.0.0.0/24 10.1.1.1 tag 3
```

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 3 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. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Configure the preferred route to a destination.

   ```
   device(config)# 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.

3. 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.

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 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.

Perform these steps to configure a default route.

1. Enter global configuration mode.

```
device# configure terminal
```

2. 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.

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
```

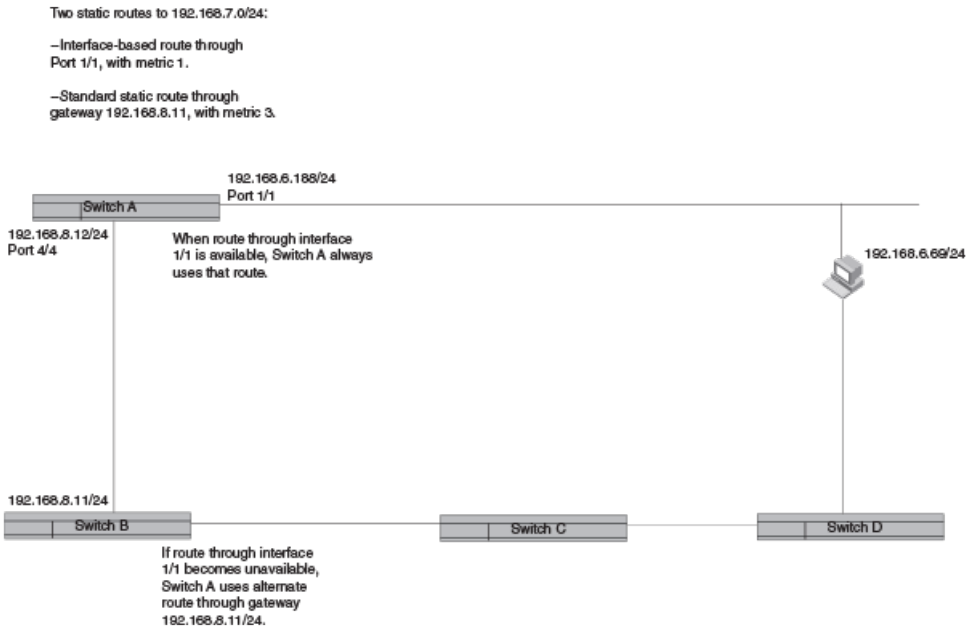
Configuring load sharing and redundancy

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 4 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 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
```

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.

3. 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
```

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#
```

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 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 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#
```

IPv6 Static Routing

- Overview of static routing....73
- Configuring a basic IPv6 static route....74
- Removing an IPv6 static route....75
- Configuring an interface as next hop....76
- Configuring a virtual interface as next hop....77
- Configuring a VRF as next hop for an IPv6 static route....78
- Adding metrics to an IPv6 static route....78
- Configuring a null route....79
- Configuring a default static route....81
- Configuring load sharing and redundancy....81
- Adding an IPv6 static route tag for use with route-maps....83
- Displaying information on IPv6 static routes....84

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 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

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:

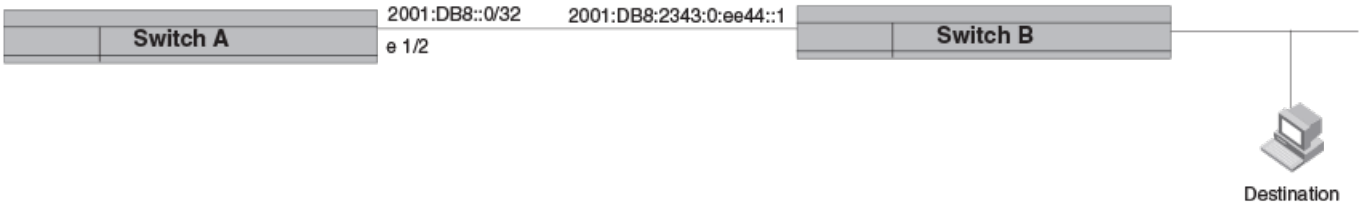
- Default routes
- Primary routes
- Backup routes
- Null routes for intentionally dropping traffic when the desired connection fails
- Alternative routes to the same destination to help load balance traffic.

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 5 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.

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.

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 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. 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
   ```

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
```

Removing an IPv6 static route

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
```

2. (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)
```

3. Enter global configuration mode.

```
device# configure terminal
```

4. 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)# no ipv6 route 2224::1/128 fe80::205:33ff:fee6:a501 ve 2
```

   This example removes the IPv6 route specified.

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#
```

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
```

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#
```

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.

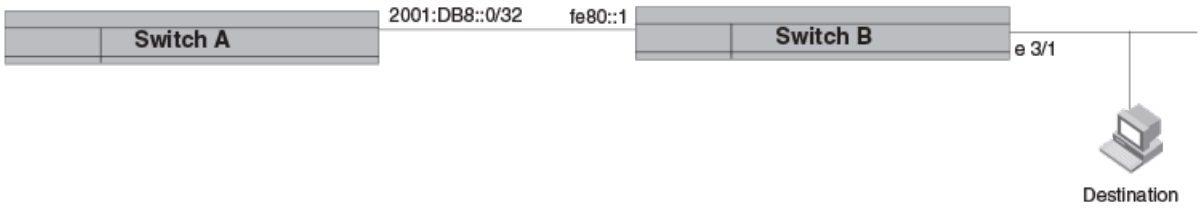
```
device# show ipv6 route 2225::/32 vrf blue
IPv6 Routing Table for VRF "blue"
Total number of IPv6 routes: 10
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

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#
```

Configuring an interface as next hop

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 6 IPv6 static route with an interface as next hop



1. Enter global configuration mode.

```
device# configure terminal
```

2. 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
```

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
```

Configuring a virtual interface as next hop

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 IP address prefix and prefix length for the route destination network.

```
device(config)# ipv6 route 2001:DB8::0/32
```

This example shows the first half of the command, the route destination, IPv6 2001:DB8::0/32 network addresses.

3. 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.

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
```

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 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 **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.

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 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. 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 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
```

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
```

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
```

Configuring a null route

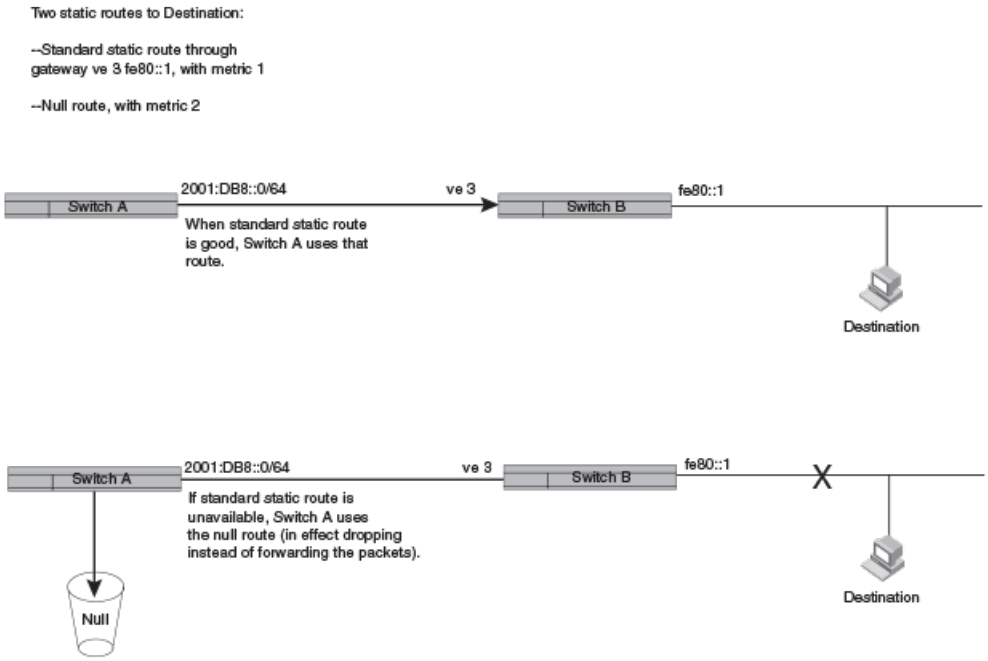
You can configure a null static route to drop packets to a certain destination. This is useful when the traffic should not be forwarded if the preferred route is unavailable.

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 7 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. Configure the preferred route to a destination.

   ```
   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.

3. Configure the null route to the same destination with a higher metric.

   ```
   device(config)# 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 fe80::1 ve 3
device(config)# 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.

Perform these steps to configure a default route.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the following destination route and network mask followed by a valid next-hop address.

```
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
```

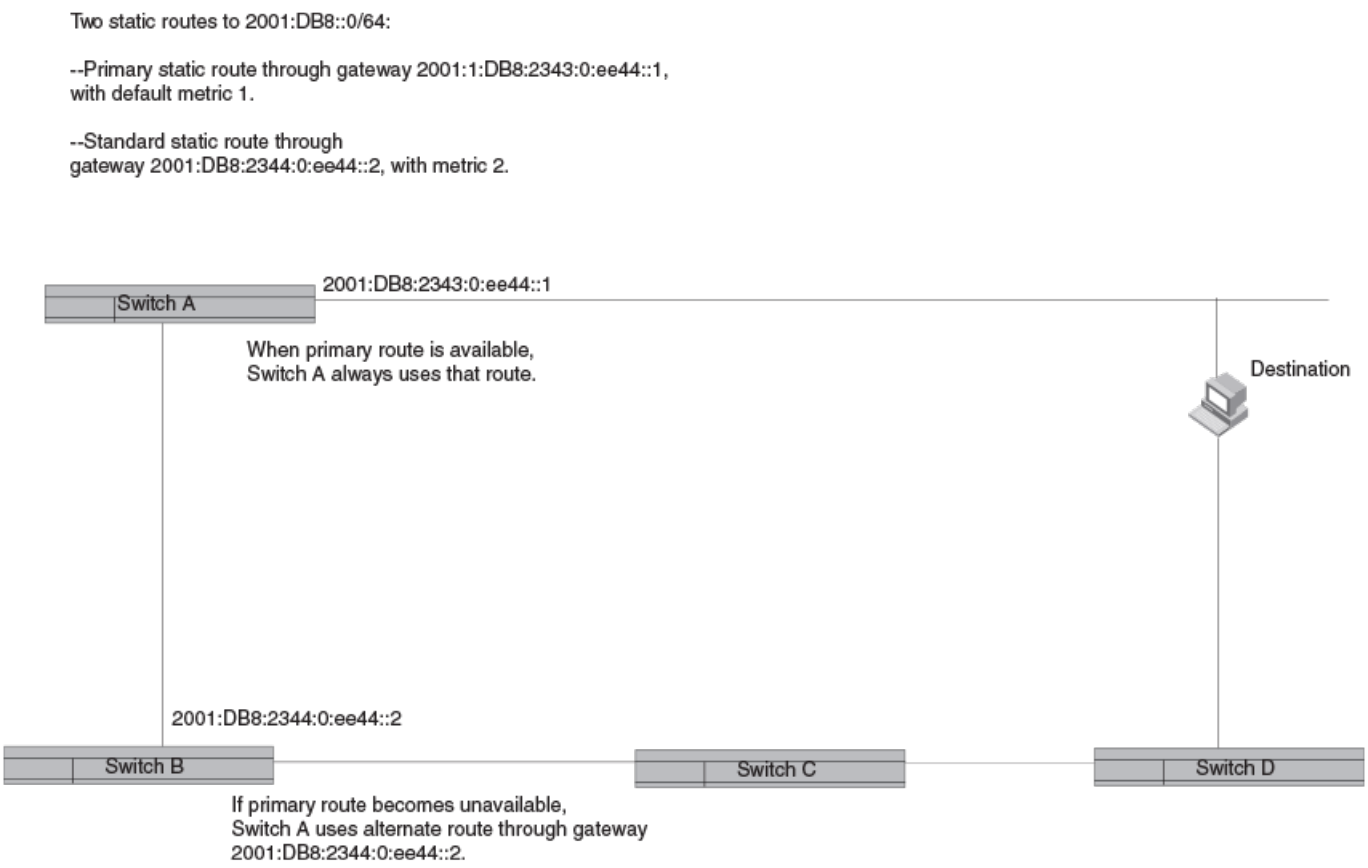
Configuring load sharing and redundancy

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 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 multiple routes to the same destination using different next hops.

```
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.

3. To prioritize multiple routes, use different metrics for each possible next hop.

```
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 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. 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)# 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)# ipv6 route 2001:DB8::0/64 2001:DB8:0:ee44::1 tag 3
```

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 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.

2. 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 85
- IP DHCP Relay function 85
- Configuring IP DHCP Relay 86
- DHCP relay agent supported user scenarios 88
- DHCP relay agent information option 82 89
- Enabling the DHCP Relay Agent Information option 90
- Configuring DHCP relay gateway address 90
- VRF support 91
- Displaying IP DHCP Relay statistics 92
- Displaying IP DHCP Relay addresses on specific devices 93
- Displaying IP DHCP relay addresses for an interface 93
- Clearing IP DHCP relay statistics 94

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 a router so that they can automatically obtain an Ethernet IP address, prefix length, and default gateway address from the DHCP server.

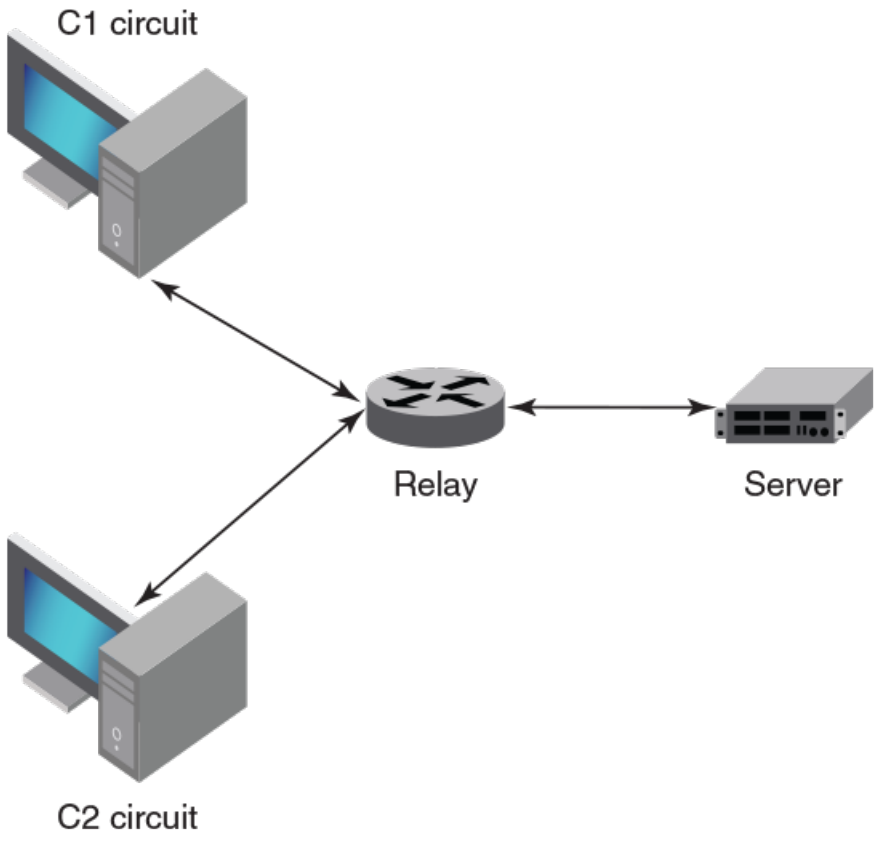
IP DHCP Relay function

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.

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 9 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), 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.
- 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 91.

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 interface configuration mode.

```
device(config)# interface ethernet 1/18
```

3. Enter the IP address of the DHCP server.

```
device(config-if-eth-1/18)# ip dhcp relay address 100.1.1.2
```

4. 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
```

5. To remove the IP DHCP Relay address enter the **no ip dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command.

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
```

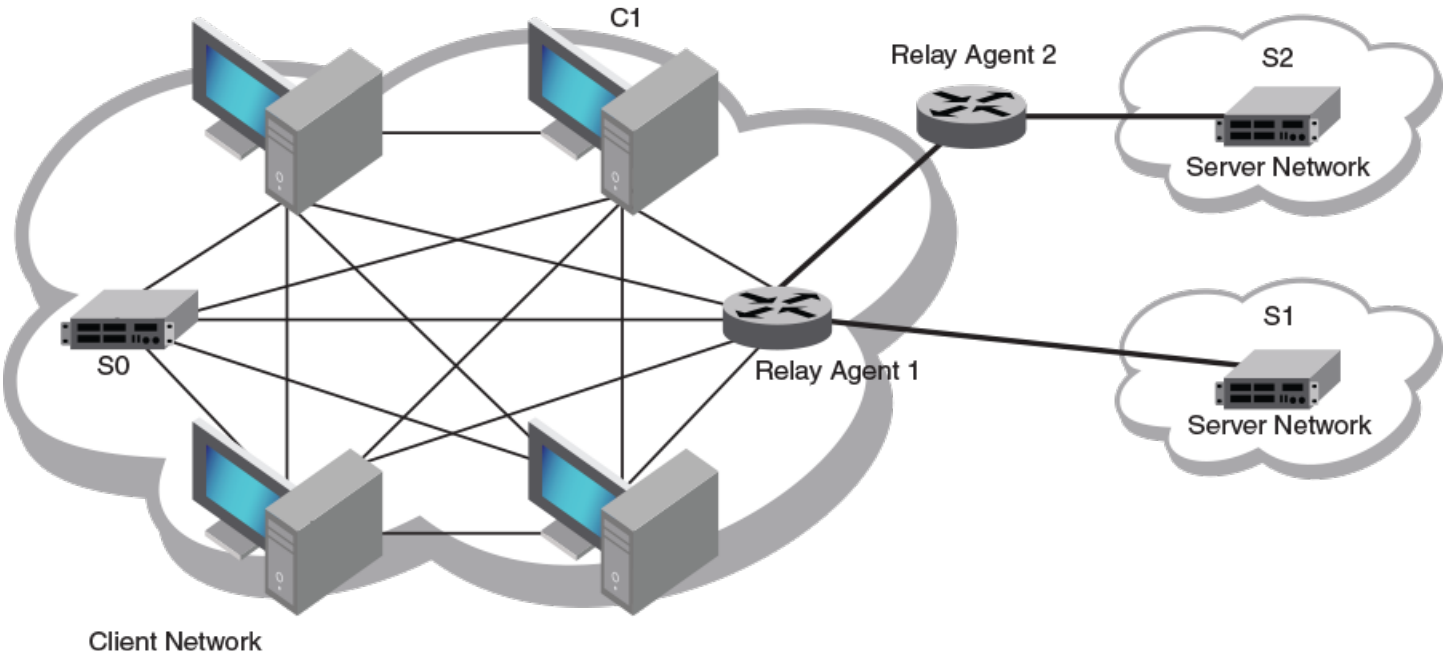
The following example removes an IP DHCP Relay address using the **no** option in the **ip dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command:

```
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 10 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.
- 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 **ip dhcp relay information option** command to enable option-82.

```
device(config)# ip dhcp relay information option
```

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 the interface configuration mode.

   ```
   device(config)# interface ethernet 1/4
   ```

3. 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.

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.

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)
- 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 <exit interface/next-hop-ip>
  - 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"

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 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.

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**.

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 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** 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
```

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 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
```

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.

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*.

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 95
- DHCP relay agent for IPv6 95
- DHCPv6 multicast addresses and UDP ports 95
- DHCPv6 address assignment 96
- DHCPv6 message format 96
- Configuring IPv6 DHCP relay 98
- Displaying DHCPv6 relay addresses on specific devices 99
- Displaying DHCPv6 relay addresses for an interface 99
- Displaying IPv6 DHCP relay statistics 100
- Clearing IP DHCPv6 relay statistics 100

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.

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 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.

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 19 DHCPv4 message versus DHCPv6 message

DHCPv6 message type	DHCPv4 message type
Solicit (1)	DHCPDISCOVER
Advertise (2)	DHCPOFFER

TABLE 19 DHCPv4 message versus DHCPv6 message (continued)

DHCPv6 message type	DHCPv4 message type
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

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.

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
```

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 **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
```

Displaying DHCPv6 relay addresses on specific devices

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** 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
```

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 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
```

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.

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**.

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 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* command.

The following command clears the IPv6 DHCP Relay address statistics globally.

```
device# clear ipv6 dhcp relay statistics
```

BFD

- Bidirectional Forwarding Detection overview....101
- General BFD considerations and limitations....102
- BFD for Layer 3 protocols....102
- BFD considerations and limitations for Layer 3 protocols....103
- BFD for Layer 3 protocols on virtual Ethernet interfaces....104
- Configuring BFD on an interface....104
- Disabling BFD on an interface....105
- BFD for BGP....105
- BFD for OSPF....109
- BFD for IS-IS....113
- Displaying BFD information....114

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 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.

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.

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.

General BFD considerations and limitations

There are a number of general points to consider when configuring BFD:

- 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 103 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 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.

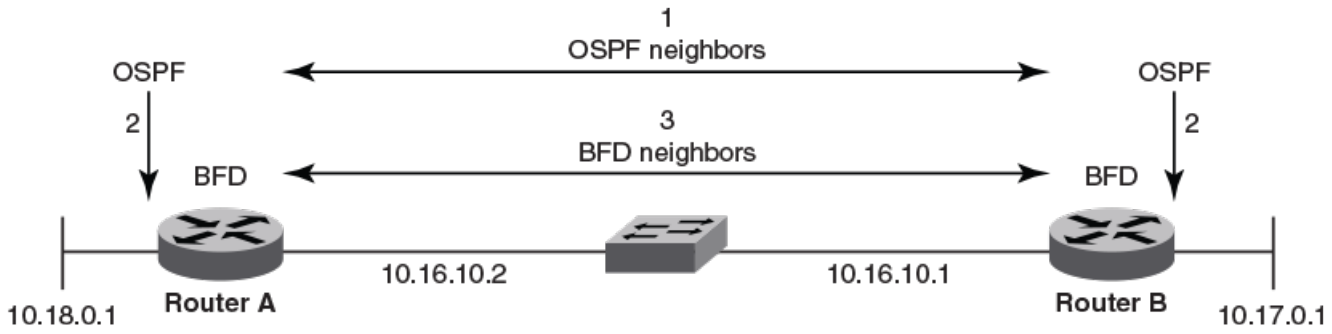
BFD provides a single point of forwarding path monitoring. This means that when more than one Layer 3 application wants to monitor a single host, BFD runs a single session for that host and provides the status to multiple applications, instead of multiple applications running individual sessions to the 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.

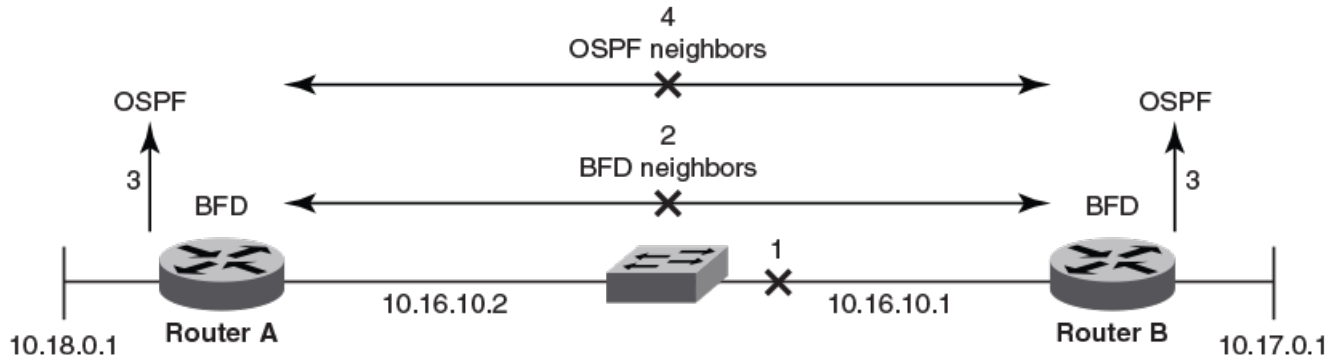
The following figure shows the establishment of a BFD session where OSPF discovers a neighbor and sends a request to BFD requesting that a BFD neighbor session be created with the OSPF neighbor router.

FIGURE 11 Establishing a BFD neighbor session



The following figure shows the termination of a BFD neighbor session after a failure occurs in the network.

FIGURE 12 Termination of a BFD neighbor session



BFD considerations and limitations for Layer 3 protocols

There are a number of things to consider when configuring BFD for Layer 3 protocols:

- 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 BGP 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.
- 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 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.

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 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-eth-1/4)# 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
```

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 ethernet 1/4
   ```

3. Enter the **bfd shutdown** command to disable BFD on the interface.

   ```
   device(config-if-eth-1/4)# bfd shutdown
   ```

The following example disables BFD on a specific Ethernet interface.

```
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.

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 **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. 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 multihop sessions only. For single-hop sessions, configuration at the interface level is considered.

4. 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)# router bgp
device(config-bgp-router)# bfd interval 110 min-rx 120 multiplier 15
device(config-bgp-router)# 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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. 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
```

4. 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.

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-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)# 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 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-router)# address-family ipv4 unicast vrf green
```

4. 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
```

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 in a nondefault VRF instance.

```
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd interval 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 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)# 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 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 **neighbor** *peer-group-name* **peer-group** command to create a peer group.

```
device(config-bgp-router)# neighbor pg1 peer-group
```

4. 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
```

5. 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
```

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-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)# 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
```

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** 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 1/4
```

3. Enter the **ip ospf bfd** command to enable BFD on the specified interface.

```
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 1/4
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 **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

3. Enter the **bfd** command to enable BFD globally.

```
device(config-router-ospf-vrf-default-vrf)# bfd
```

4. Enter the **bfd holdover-interval** command to set the BFD holdover interval.

```
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-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 **router ospf** command and specify a VRF name to enable OSPFv2 in a nondefault VRF instance.

   ```
   device(config)# router ospf vrf red
   ```

3. Enter the **bfd** command to enable BFD.

   ```
   device(config-router-ospf-vrf-red)# bfd
   ```

4. 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)# 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-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-if-Ve-24)# 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 **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3.

```
device(config)# ipv6 router ospf
```

3. Enter the **bfd** command to enable BFD globally.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# bfd
```

4. Enter the **bfd holdover-interval** command to set the BFD holdover interval.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# bfd holdover-interval 20
```

The following example enables BFD and sets the BFD holdover interval to 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 **ipv6 router ospf** command and specify a VRF name to enable OSPFv3 in a nondefault VRF instance.

```
device(config)# ipv6 router ospf vrf orange
```

3. Enter the **bfd** command to enable BFD.

```
device(config-ipv6-router-ospf-vrf-orange)# bfd
```

4. 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)# 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-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-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 to enable BFD globally.

```
device(config-isis-router)# bfd
```

4. 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
device(config-isis-router)# bfd holdover-interval 12
```

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.

BGP4

- BGP4 overview 118
- BGP4 peering 119
- BGP4 message types 119
- BGP4 attributes 121
- BGP4 best path selection algorithm 121
- Device ID 123
- BGP global mode 123
- Configuring a local AS number 124
- IPv4 unicast address family 124
- Neighbor configuration 125
- Peer groups 126
- Advertising the default BGP4 route 127
- Four-byte AS numbers 128
- Cooperative BGP4 route filtering 129
- BGP4 parameters 130
- Route redistribution 131
- Advertised networks 131
- Importing routes into BGP4 132
- Static networks 132
- Route reflection 133
- Route flap dampening 134
- Aggregating routes advertised to BGP neighbors 135
- Advertising the default BGP4 route 135
- Advertising the default BGP4 route to a specific neighbor 136
- Multipath load sharing 136
- Specifying the weight added to received routes 137
- Using the IPv4 default route as a valid next hop for a BGP4 route 137
- Adjusting defaults to improve routing performance 138
- Next-hop recursion 138
- Route filtering 139
- BGP regular expression pattern-matching characters 139
- Timers 140
- Enabling BGP4 in a non-default VRF 140
- BGP4 outbound route filtering 141
- BGP4 confederations 143
- BGP community and extended community 145
- BGP4 graceful restart 147
- BGP additional-paths 149
- BGP best-external route 158
- Auto shutdown of BGP neighbors on initial configuration 160
- Generalized TTL Security Mechanism support 161
- Disabling the BGP AS_PATH check function 163
- Using route maps 163
- Matching on an AS-path 164
- Matching on a community ACL 165
- Matching on a destination network 165
- Matching on a BGP4 static network 166
- Matching on a next-hop device 167

- Matching on an interface....167
- Using route-map continue statements....167
- Route-map continue statement for BGP4 routes....169
- Using a route map to configure dampening....169
- Clearing diagnostic buffers....169
- Displaying BGP4 statistics....170
- Displaying BGP4 neighbor statistics....173
- Clearing BGP4 dampened paths....175

BGP4 overview

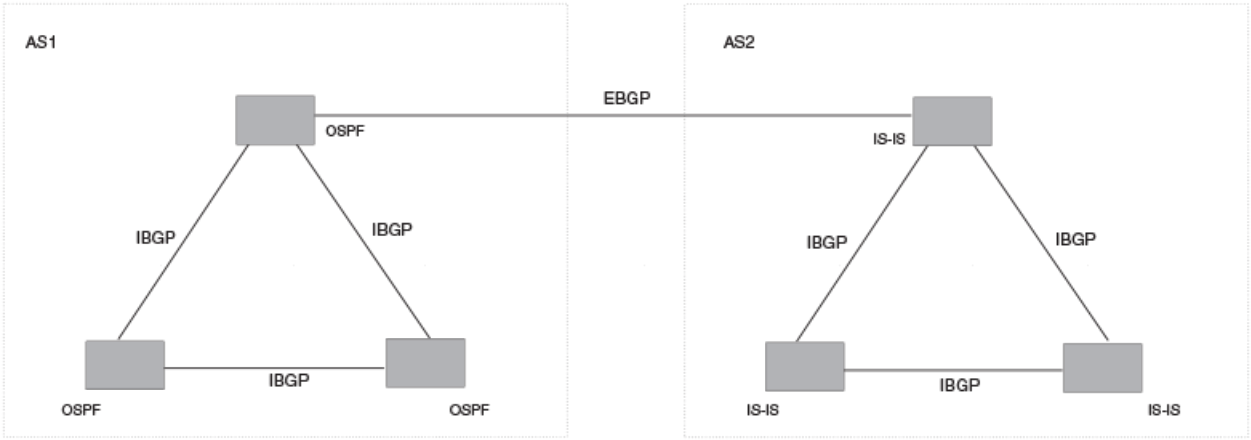
Border Gateway Protocol version 4 (BGP4) is an exterior gateway protocol that performs inter-autonomous system (AS) or inter-domain routing. It peers to other BGP-speaking systems over TCP to exchange network reachability and routing information. BGP primarily performs two types of routing: inter-AS 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 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.

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 13 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 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 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.

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 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
  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
```

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 **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
```

The following example configures the local ASN for a device.

```
device# configure terminal
device(config)# 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-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
```

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 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
```

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
```

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 **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 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
   ```

The following example configures a BGP4 neighbor.

```
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
```

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 **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** *peer-group-name* **peer-group** command to create a peer group.

```
device(config-bgp-router)# 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-router)# 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-router)# neighbor 10.2.2.2 peer-group mypeergroup1
```

7. 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
```

The following example creates a peer group and specifies two neighbors to belong to the peer group.

```
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
```

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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv4 unicast** command to enter IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

4. 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
```

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)# 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.
- 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, ISIS, and OSPF routes into BGP is supported. Similarly, routes learned through BGP can also be redistributed into OSPF or IS-IS.

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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv4 unicast** command to enter BGP IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

4. Enter the **redistribute** command using the **connected** keyword to redistribute connected routes.

```
device(config-bgp-ipv4u)# redistribute connected
```

The following example redistributes connected routes into BGP4.

```
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, ISIS, 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.

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 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.

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 **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
```

4. Enter the **address-family ipv4 unicast** command to enter address family IPv4 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **network** command and specify a *network/mask* to import the specified prefix into the BGP4 database.

```
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)# 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
```

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.

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 **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 **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. 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
```

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
```

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 **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 **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
```

The following example changes the cluster ID of a device from the default device ID to 321.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# 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 **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 **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. 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
```

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-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.

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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv4 unicast** command to enter BGP address family IPv4 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

4. 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
```

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)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# 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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv4 unicast** command to enter IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

4. 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
```

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)# 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 **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 **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

5. 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
   ```

The following example enables a BGP4 device to advertise the default IPv4 route to a specific neighbor.

```
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.

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 **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 65520
```

4. Enter the **address-family ipv4 unicast** command to enter address family IPv4 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. 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-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)# 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
```

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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv4 unicast** command to enter IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

4. Enter the **next-hop-enable-default** command to configure the device to use the default route as a valid next hop.

```
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)# 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-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-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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv4 unicast** commandto enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

4. Enter the **next-hop-recursion** command to enable recursive next hop lookups.

```
device(config-bgp-ipv4u)# next-hop-recursion
```

The following example enables recursive next hop lookups.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# next-hop-recursion
```

Route filtering

The following route filters are supported:

- AS-path filter
- Community filter
- Prefix list
- Route map
- Table map

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 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 20 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

TABLE 20 BGP regular expression pattern-matching characters (continued)

Regular expression character	Function	Examples
^	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.

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 **router bgp** command to enable BGP routing globally.

```
device(config)# router bgp
```

3. 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
```

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)# router bgp
device(config-bgp-router)# 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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

4. 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-ipv4u)# neighbor 10.1.2.3 prefix-list myprefixlist in
```

5. 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-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-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-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist
     ```

The following example configures ORF in receive mode.

```
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)# 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)# 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.

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

   ```
   device# configure terminal
   ```

2. Enter the **ip prefix-list** command to to configure the IP prefix list instance.

   ```
   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
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   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 **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-ipv4u)# neighbor 10.2.3.4 prefix-list Routesfrom1234 in
```

6. Enter the **capability orf prefixlist** command with the **send** parameter to enable the ORF prefix list capability in send mode.

```
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-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
```

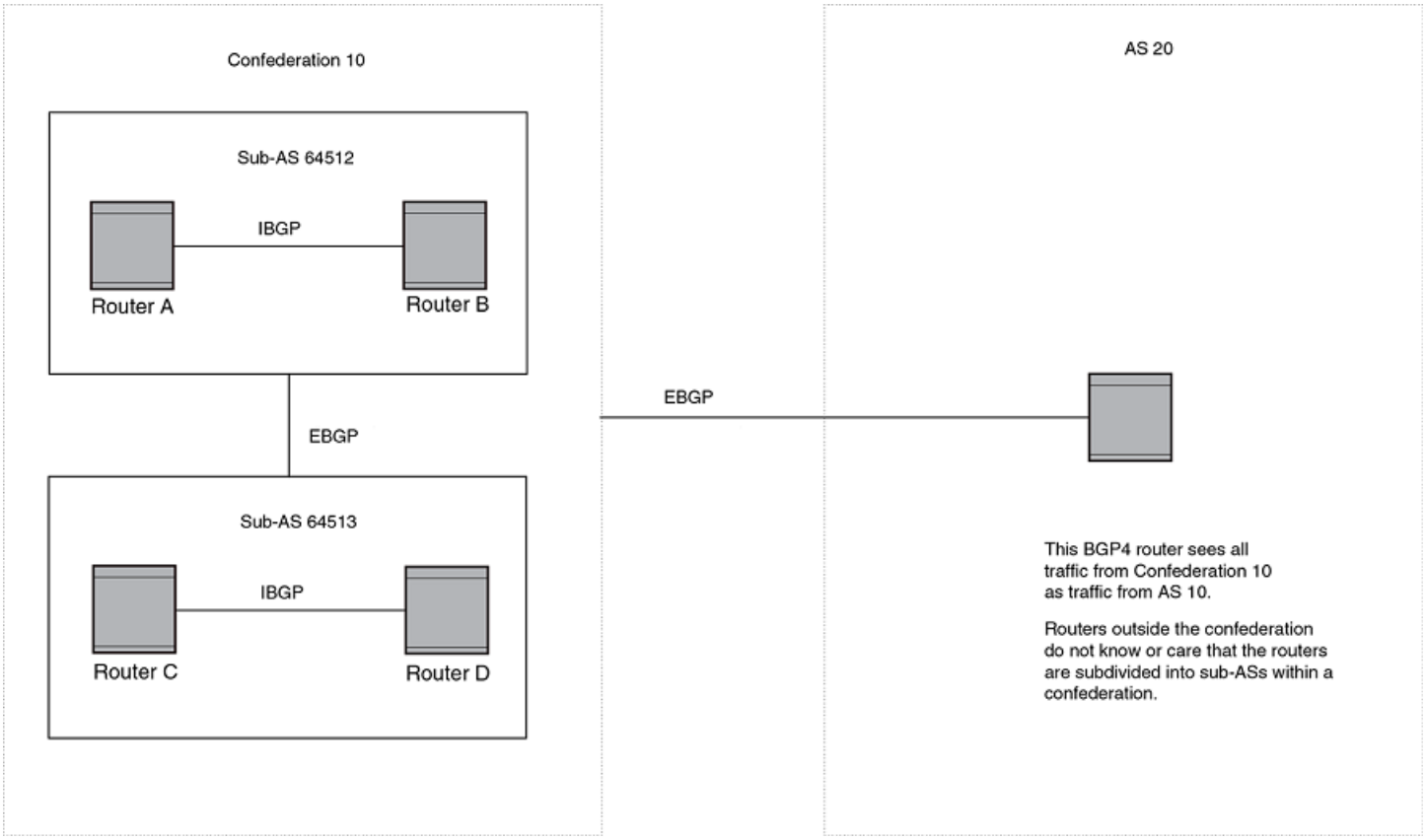
BGP4 confederations

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 14 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 **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 65520
```

4. Enter the **confederation identifier** command and specify an ASN to configure a BGP confederation identifier.

```
device(config-bgp-router)# confederation identifier 100
```

5. 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
```

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 **ip extcommunity-list** command and specify a number to set a BGP extended community filter.

```
device(config)# ip extcommunity-list 1 permit soo 123:2
```

3. Enter the **route-map** *name* command to create and define a route map and enter route-map configuration mode.

```
device(config)# route-map extComRmap permit 10
```

   Permits a matching pattern.

4. Enter the **match extcommunity** command and specify an extended community list number.

```
device(config-route-map-extComRmap/permit/10)# match extcommunity 1
```

5. Enter the **exit** command.

```
device(config-route-map-extComRmap/permit/10)# exit
```

6. Enter the **route-map***name* command to define a route map and enter route-map configuration mode.

```
device(config)# route-map sendExtComRmap permit 10
```

   Permits a matching pattern.

7. 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
```

8. 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)# 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 **ip extcommunity-list** command and specify a number to set a BGP extended community filter.

```
device(config)# ip extcommunity-list 1 permit rt 123:2
```

3. Enter the **router bgp** command to enable BGP routing.

```
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
```

5. 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
```

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 **neighbor** *ip-address* **activate** command to enable the exchange of information with the neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 activate
```

8. 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
```

9. 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
```

10. 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
```

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-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
```

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.

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 **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** *ip address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
```

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 **graceful-restart** command to enable the graceful restart feature.

```
device(config-bgp-ipv4u)# graceful-restart
```

7. 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
```

   - 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
```

   - 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
```

The following example enables the GR feature.

```
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-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-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-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 additional-paths

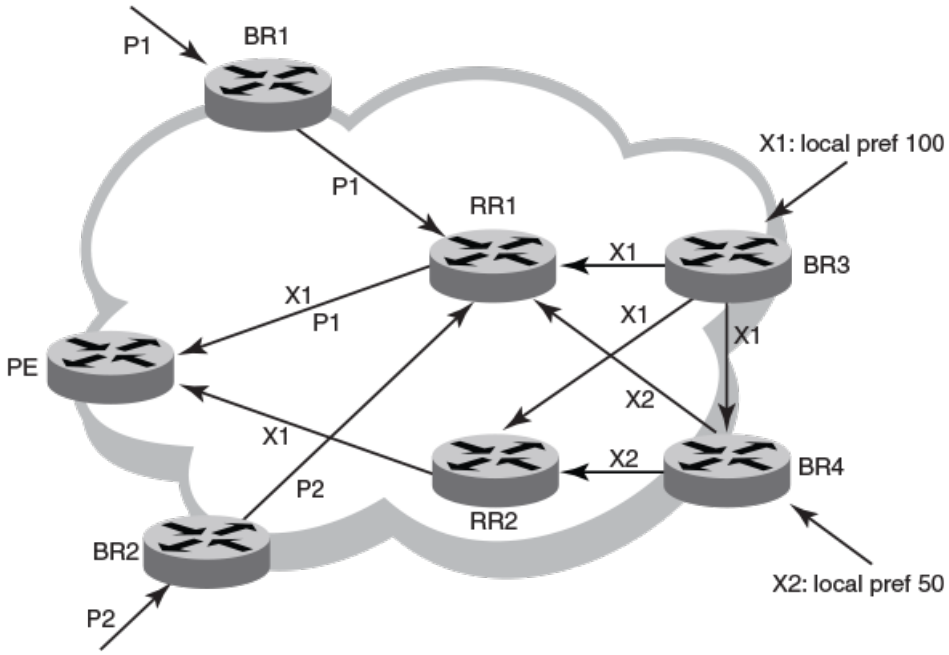
BGP additional-paths enables the advertisement of multiple paths for the same prefix without new paths implicitly replacing the previous paths. Path diversity is achieved rather than path hiding.

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 15 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.

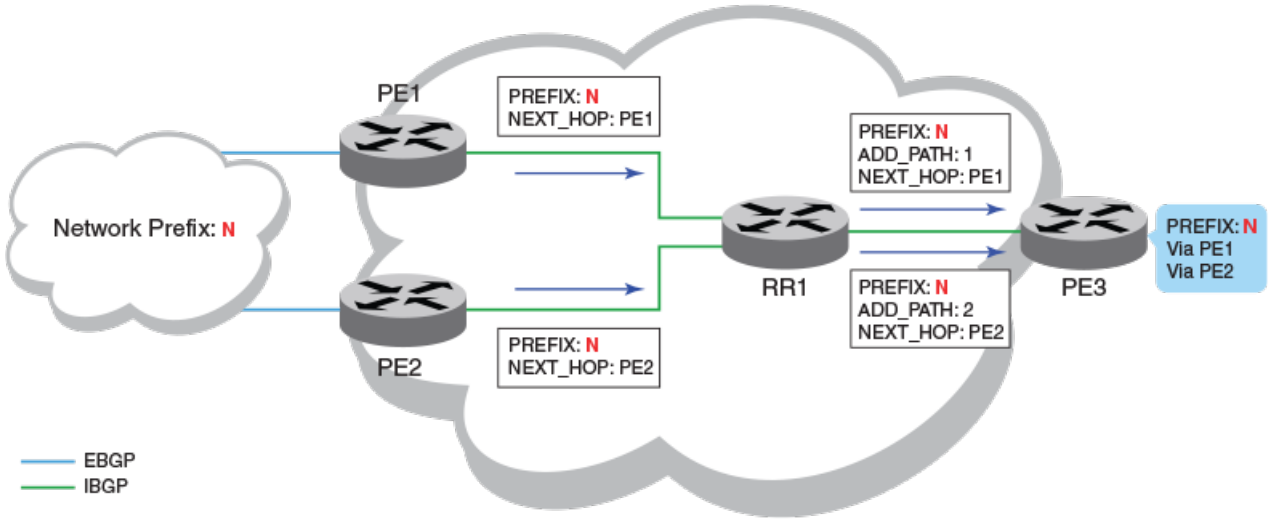
When the additional-paths capability is negotiated, BGP updates carry the path ID. To carry the path ID in an update message, the existing NLRI encodings are extended by prepending the path ID field, which consists of four octets.

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 16 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.
- 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.

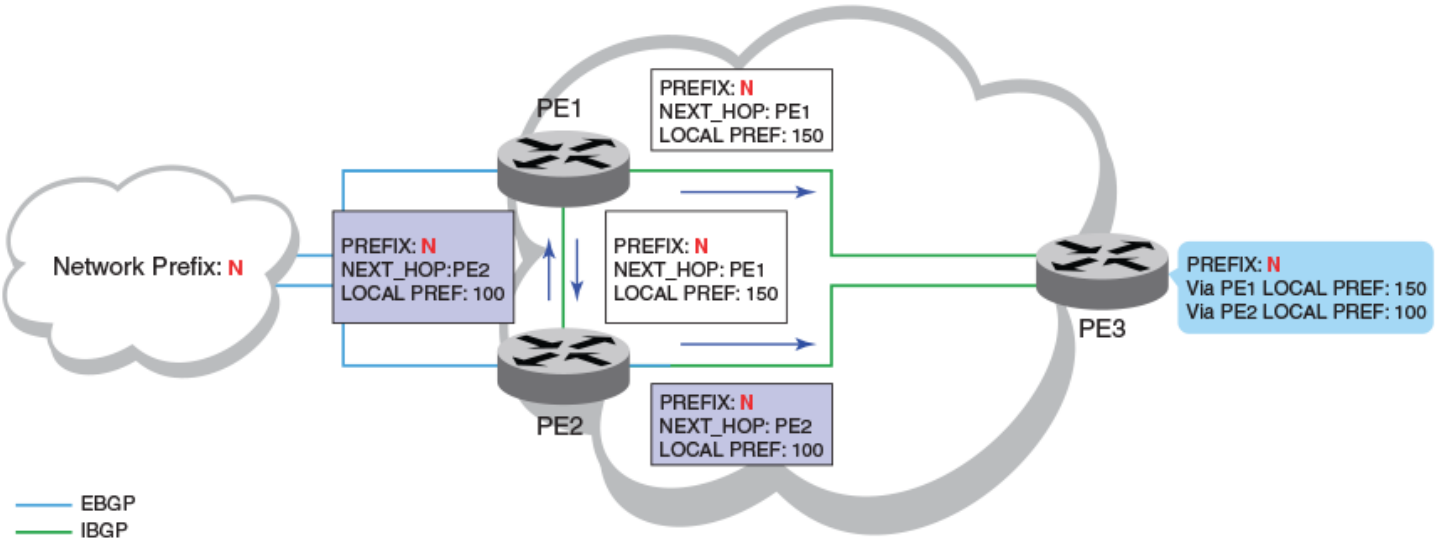
The best external route, when different from the best route, acts as a backup route and introduces additional information into an Interior Border Gateway Protocol (iBGP) mesh, which may be useful in restoration of connectivity.

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 17 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 **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 65520
```

4. 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-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-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 **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 65520
```

4. 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-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
```

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.

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 **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 65520
```

4. Enter the **neighbor remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
```

5. Enter the **neighbor ebgp-btsh** command, specifying an IP address, to enable GTSM.

```
device(config-bgp-router)# 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
```

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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv4 unicast** command to enter BGP IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

4. 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-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)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# 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 (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.

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 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 **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
```

3. Enter the **match** command and specify an ACL name to configure the route map to match on the specified AS-path ACL.

```
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 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 **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
```

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 mycommroutemap1 permit 10
```

4. 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/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/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 **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
```

3. 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/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/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 **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
```

3. Enter the **match** command with the **protocol bgp static-network** parameter to match on BGP4 static network routes.

```
device(config-routemap-mystaticroutemap3/permit/1)# match protocol bgp static-network
```

4. 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/permit/1)# set local-preference 150
```

5. 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/permit/1)# set community no-export
```

6. Enter the **exit** command to exit route map configuration mode.

```
device(config-routemap-mystaticroutemap3/permit/1)# exit
```

7. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

8. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

9. 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-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/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 **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
   ```

3. 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-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/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)# route-map myintroutemap1 permit 99
device(config-route-map-myintroutemap1/permit/99)# match interface ethernet 1/1 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 **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
```

3. Enter the **match** command with the metric parameter and specify a value to match a route metric in the route-map instance.

```
device(config-route-map-mcontroutemap1/permit/1)# match metric 10
```

4. 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-route-map-mcontroutemap1/permit/1)# set weight 10
```

5. 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-mycontroutemap/permit/1)# continue 2
```

6. Enter the **exit** command to exit route map configuration mode.

```
device(config-routemap-mycontroutemap/permit/1)# exit
```

7. 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
```

8. Enter the **match** command with the metric parameter and specify a value to match a route metric in the route-map instance.

```
device(config-route-map-mcontroutemap1/permit/2)# match metric 10
```

9. 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-route-map-mcontroutemap1/permit/2)# set weight 20
```

10. 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-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-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 **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
   ```

3. 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/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 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 *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: 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 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.

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
```

BGP4+

- BGP4+ overview 177
- BGP global mode 178
- IPv6 unicast address family 179
- BGP4+ neighbors 180
- BGP4+ peer groups 182
- Enabling ASN capability globally 184
- Enabling ASN capability for a BGP4+ neighbor 185
- Importing routes into BGP4+ 185
- Advertising the default BGP4+ route 186
- Advertising the default BGP4+ route to a specific neighbor 186
- Using the IPv6 default route as a valid next hop for a BGP4+ route 187
- BGP4+ next hop recursion 188
- BGP4+ NLRIs and next hop attributes 189
- BGP4+ route reflection 189
- BGP4+ route aggregation 190
- BGP4+ multipath 191
- Route maps 192
- Redistributing prefixes into BGP4+ 194
- Redistributing routes into BGP4+ 194
- Specifying the weight added to BGP4+ received routes 195
- Enabling BGP4+ in a non-default VRF 195
- BGP4+ outbound route filtering 196
- BGP4+ confederations 197
- BGP4+ extended community 198
- BGP4+ graceful restart 201
- Auto shutdown of BGP neighbors on initial configuration 203
- Generalized TTL Security Mechanism support 204
- Disabling the BGP AS_PATH check function 205
- Displaying BGP4+ statistics 206
- Displaying BGP4+ neighbor statistics 208
- Clearing BGP4+ dampened paths 210

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)
- 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.

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
  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
```

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
  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
```

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.

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.

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
```

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
  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 **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 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
```

5. Enter the **address family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

6. 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)# 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
```

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
```

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
```

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
```

6. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# 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-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-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
```

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 **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 peer-group** command to create a peer group.

```
device(config-bgp-router)# neighbor mypeergroup1 peer-group
```

5. 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
```

6. 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
```

7. 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
```

8. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

9. 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)# 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
```

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 **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 peer-group** command, specifying a peer group, to create a peer group.

```
device(config-bgp-router)# neighbor p1 peer-group
```

5. Enter the **neighbor remote-as** command to specify the ASN of the peer group.

```
device(config-bgp-router)# neighbor p1 remote-as 11
```

6. 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
```

7. 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
```

8. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

9. 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)# 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
```

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.

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 **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
```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

5. 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)# 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
```

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 **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **address-family ipv6 unicast** command to enter IPv6 address family unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

4. 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)# router bgp
device(config-bgp-router)# 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 **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 **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

5. 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)# 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
```

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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

4. 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)# router bgp
device(config-bgp-router)# 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 **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

4. 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)# router bgp
device(config-bgp-router)# 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 **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 **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
```

The following example changes the cluster ID of a device from the default device ID to 321.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# 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 **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 **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

5. 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)# 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
```

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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

4. 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)# router bgp
device(config-bgp-router)# 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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

4. 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
```

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
```

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/permit/10)# match ipv6 address prefix-list myprefixlist
```

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

```
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
```

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
```

9. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# 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/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, 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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

4. 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
```

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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv6 unicast** command to enter IPv6 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

4. 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
```

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 **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 65520
```

4. Enter the **address-family ipv6 unicast** command to enter address family IPv6 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

5. 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)# 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
```

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 **router bgp** command to enable BGP routing globally.

```
device(config)# router bgp
```

3. 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
```

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)# router bgp
device(config-bgp-router)# 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 **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

4. Enter the **neighbor activate** command, specifying an IPv6 address, to add a neighbor.

   ```
   device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
   ```

5. 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
   ```

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-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)# 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
```

The following example configures ORF in send mode.

```
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
```

The following example configures ORF in both send and receive modes.

```
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
```

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 **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 65520
```

4. Enter the **confederation identifier** command and specify an ASN to configure a BGP confederation identifier.

```
device(config-bgp-router)# confederation identifier 100
```

5. 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
```

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 **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
   ```

3. Enter the **route-map** *name* command to create and define a route map and enter route-map configuration mode.

   ```
   device(config)# route-map extComRmap permit 10
   ```

   Permits a matching pattern.

4. Enter the **match extcommunity** command and specify an extended community list.

   ```
   device(config-route-map-extComRmap/permit/10)# match extcommunity mylist
   ```

5. Enter the **exit** command.

   ```
   device(config-route-map-extComRmap/permit/10)# exit
   ```

6. Enter the **route-map***name* command to define a route map and enter route-map configuration mode.

   ```
   device(config)# route-map sendExtComRmap permit 10
   ```

   Permits a matching pattern.

7. 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
   ```

8. 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)# 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 **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
```

3. Enter the **router bgp** command to enable BGP routing.

```
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
```

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 fe80:4398:ab30:45de::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
```

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 **in** keyword to apply a route map to incoming routes.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map in extComRmapt
```

9. 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
```

10. 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 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.

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 **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 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
```

5. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

6. Enter the **neighbor activate** command to add a neighbor.

```
device(config-bgp-ipv6u)# neighbor 1000::1 activate
```

7. Enter the **graceful-restart** command to enable the graceful restart feature.

```
device(config-bgp-ipv6u)# graceful-restart
```

8. 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)# 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)# 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)# 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)# 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 **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 65520
   ```

4. 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-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-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 **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 65520
```

4. 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-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-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.

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 **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 65520
```

4. Enter the **neighbor remote-as** command, specifying an IPv6 address, to add a neighbor.

```
device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
```

5. Enter the **neighbor ebgp-btsh** command, specifying an IPv6 address, to enable GTSM.

```
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-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 **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv6 unicast** command to enter IPv6 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

4. 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
```

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+ 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.

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....213
- BGP EVPN control plane....216
- BGP EVPN configuration examples....217
- BGP routing tables....222
- BGP EVPN-based MCT cluster formation....223
- BGP EVPN-based VXLAN overlay....225
- EVPN next-hop resolution and tunnel-EVI membership....230
- Layer 2 (MAC) route exchange....231
- ARP/ND (MACIP) route exchange....235
- EVPN prefix route exchange and Multi-VRF support....236
- BGP EVPN VXLAN data center interconnect....239
- Static anycast gateway....241
- IP unnumbered interface....243
- High availability....244

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.

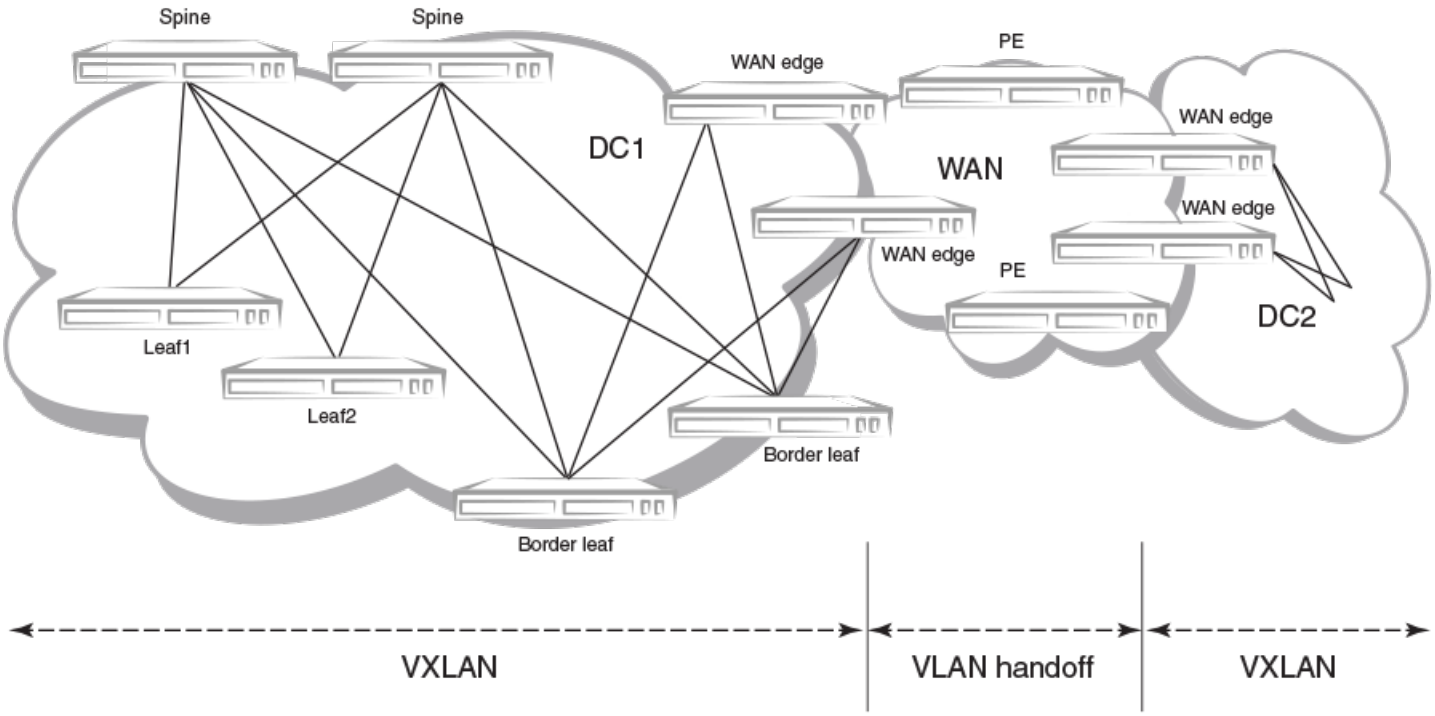
BGP EVPN for VXLAN overlay networks, along with a broad set of Layer 2, Layer 3, and Infrastructure features to enable the seamless deployment, on-demand usage of forwarding entries in hardware, and minimization of flooding in the network, are referred to as a BGP EVPN-based IP Fabric. This feature is 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

The following figure shows a supported topology where the SLX 9850 or SLX 9540 act as nodes in a BGP EVPN spine in an IP Fabrics network.

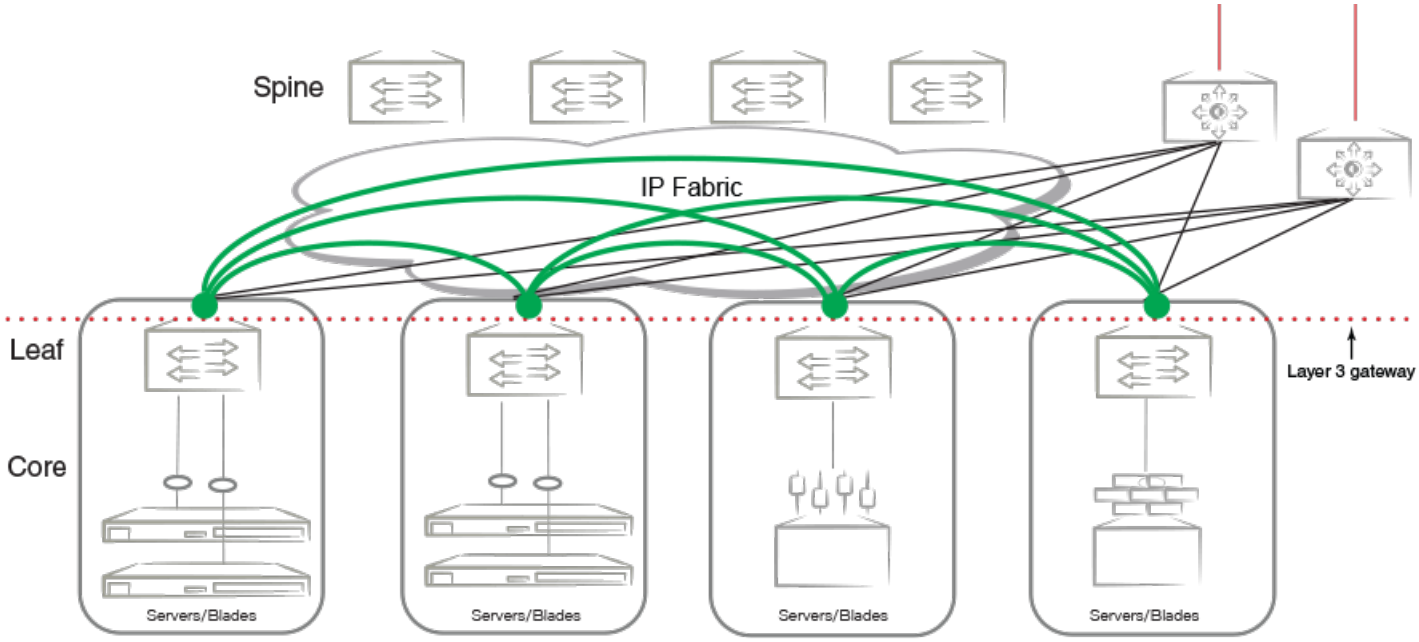
FIGURE 18 Supported BGP EVPN spine topology



The leaf nodes in the IP Fabric can be VDX series or SLX 9140 and SLX 9540 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 19 VXLAN connectivity in an IP Fabric



NOTE
Leaf nodes will be dual-homed.

IP Fabrics features

This release supports the following IP Fabrics features:

- BGP EVPN spine with route reflector control plane
- VXLAN VTEP discovery
- VXLAN logical VTEP
- BGP EVPN-based MCT cluster formation
- Layer 2 (MAC) route exchange
- ARP/ND (MACIP) route exchange
- Static anycast gateway
- ARP suppression
- IP unnumbered interfaces
- Multi-VRF support and EVPN prefix route exchange
- Conversion of MACIP routes to host prefix routes

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 (Layer 2 services on the SLX 9540 only)

BGP EVPN control plane

This section discusses the following components of the BGP EVPN control plane.

Supported route types

The following table describes the supported EVPN route types.

TABLE 21 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	IPv4/IPv6 Prefix Route
7	IGMP Join Synch Route
8	IGMP Leave Synch 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
- 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:

- MPLS (for MCT support)
- 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
```

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
```

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.

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
```

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
```

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
```

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
```

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
```

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
```

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
```

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 22 Calculating EVIDs for VLANBs/BDs

VLAN/BD	EVI value
VLAN (1–4095)	VLANID
BD (1–4096+)	4096 + BD-ID

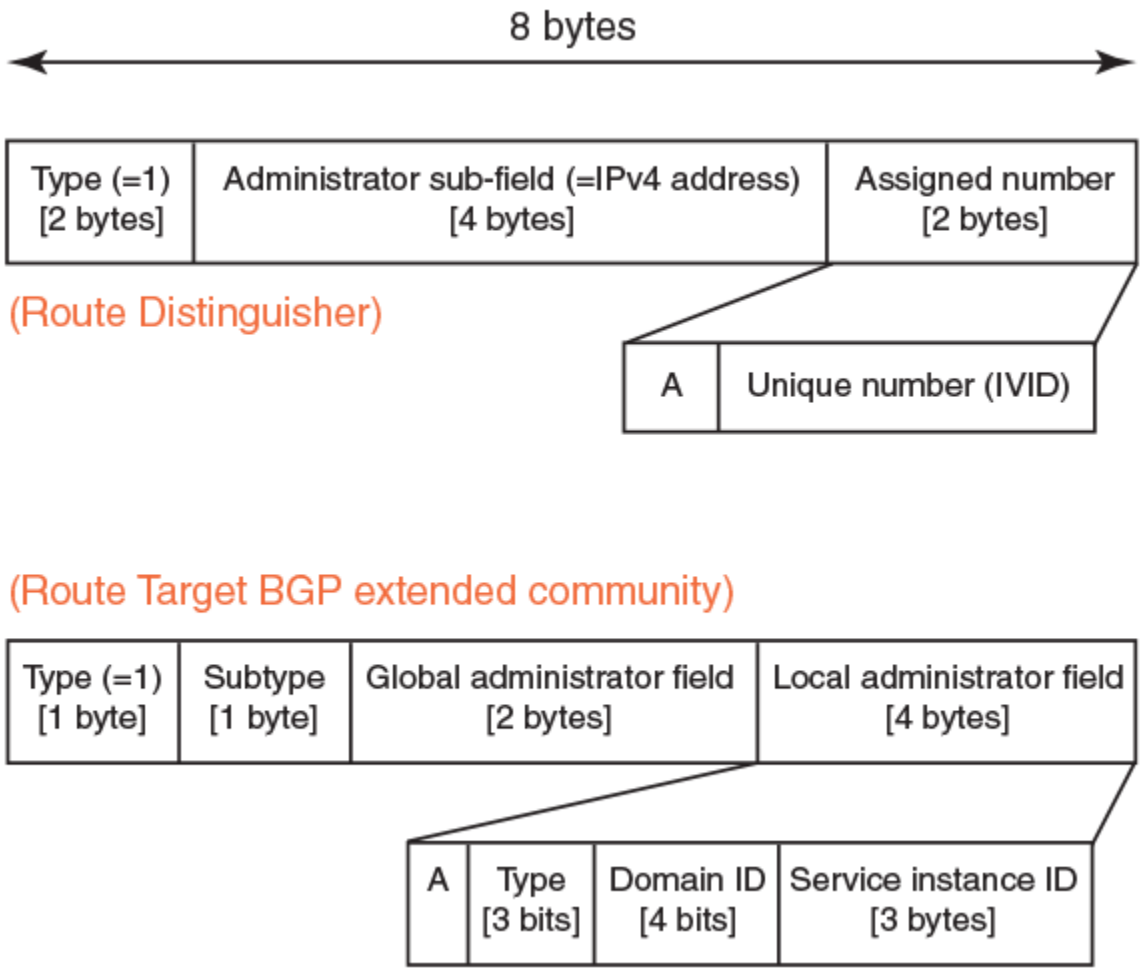
The EVID value in the preceding configuration example is not sent in the Ethernet-tag field in the routes. It is always 0 for VLAN-aware service. However, routes imported into the MAC-VRF table are classified against an EVID, which is shown in the Ethernet-tag field.

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 20 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 23 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.

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 24 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 25 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)

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.

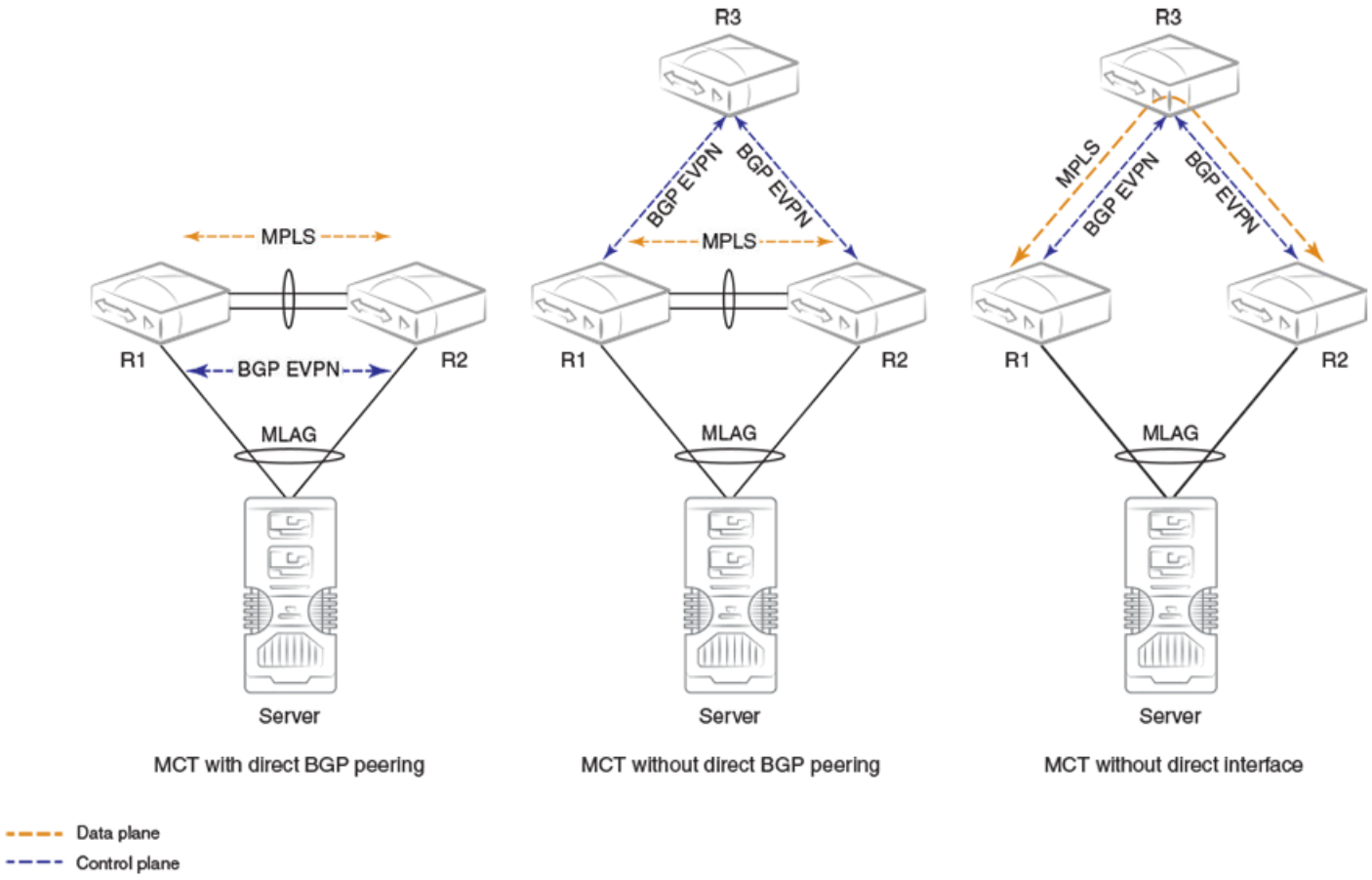
BGP EVPN peering for the MCT cluster

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 21 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 26 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.

Devices in the underlay Layer 3 Clos network may have the following distinct roles:

- Leaf
- Spine
- Border leaf
- Colocated border leaf and spine

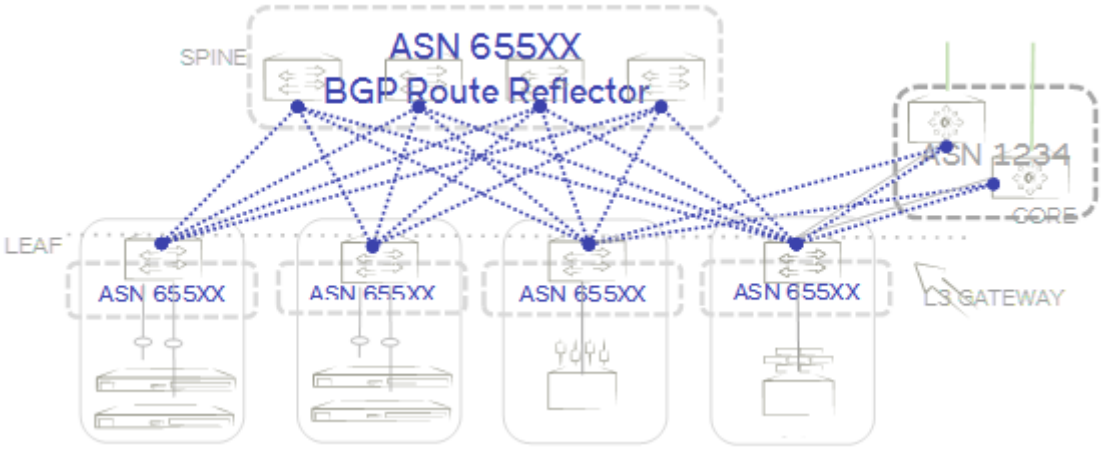
Underlay architectures

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.

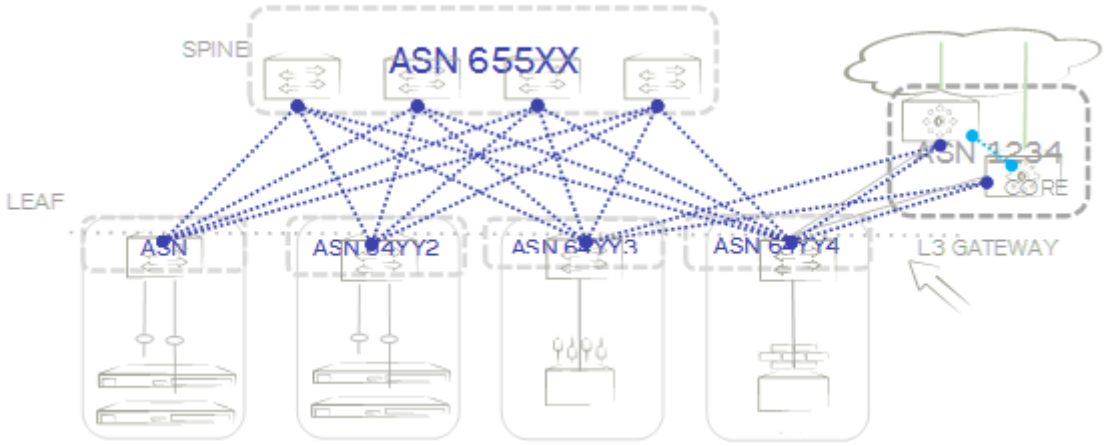
FIGURE 22 iBGP spine and leaf



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 23 eBGP spine and leaf

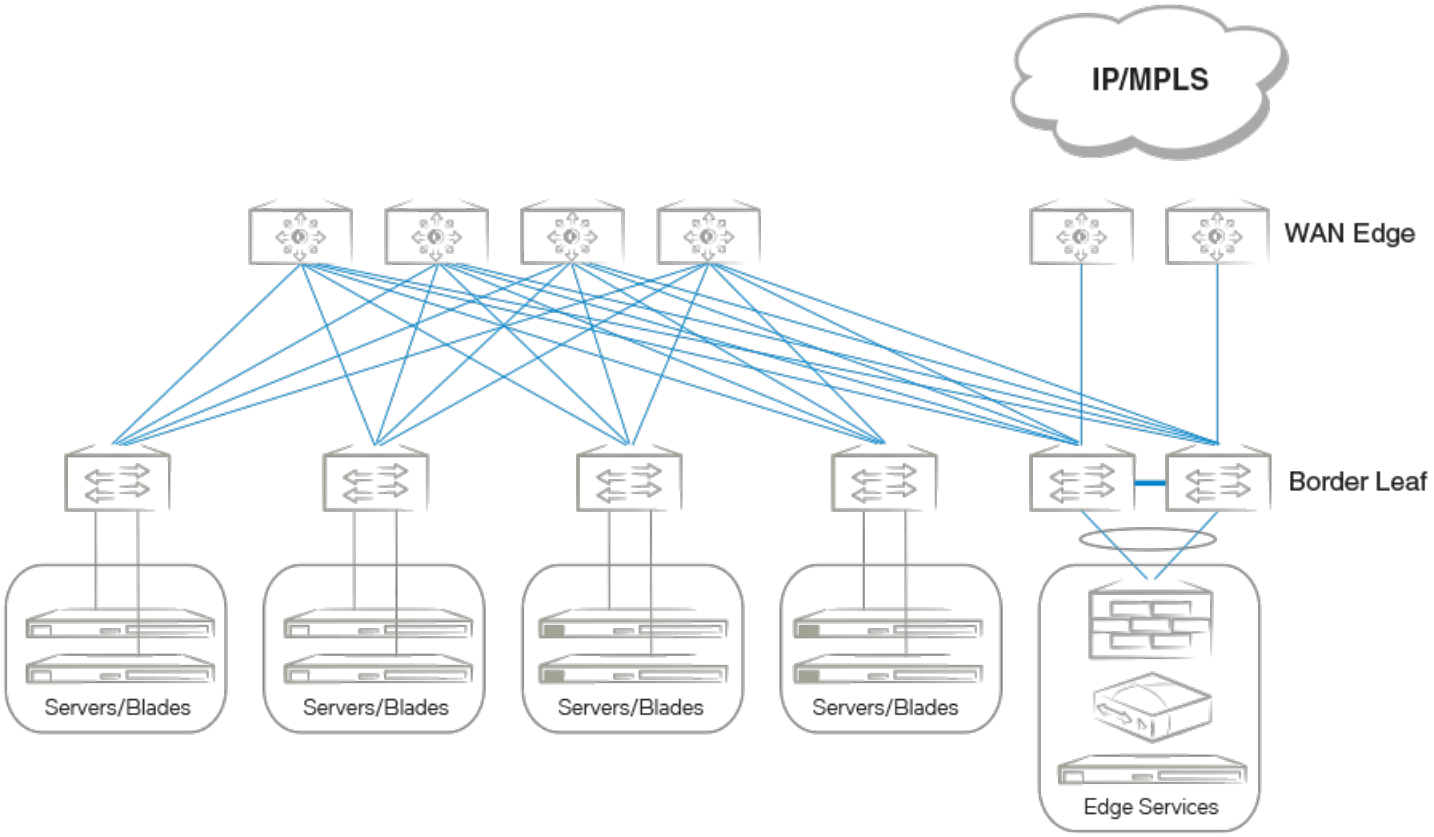


Border and service leaf

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 24 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
!
```

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
!
```

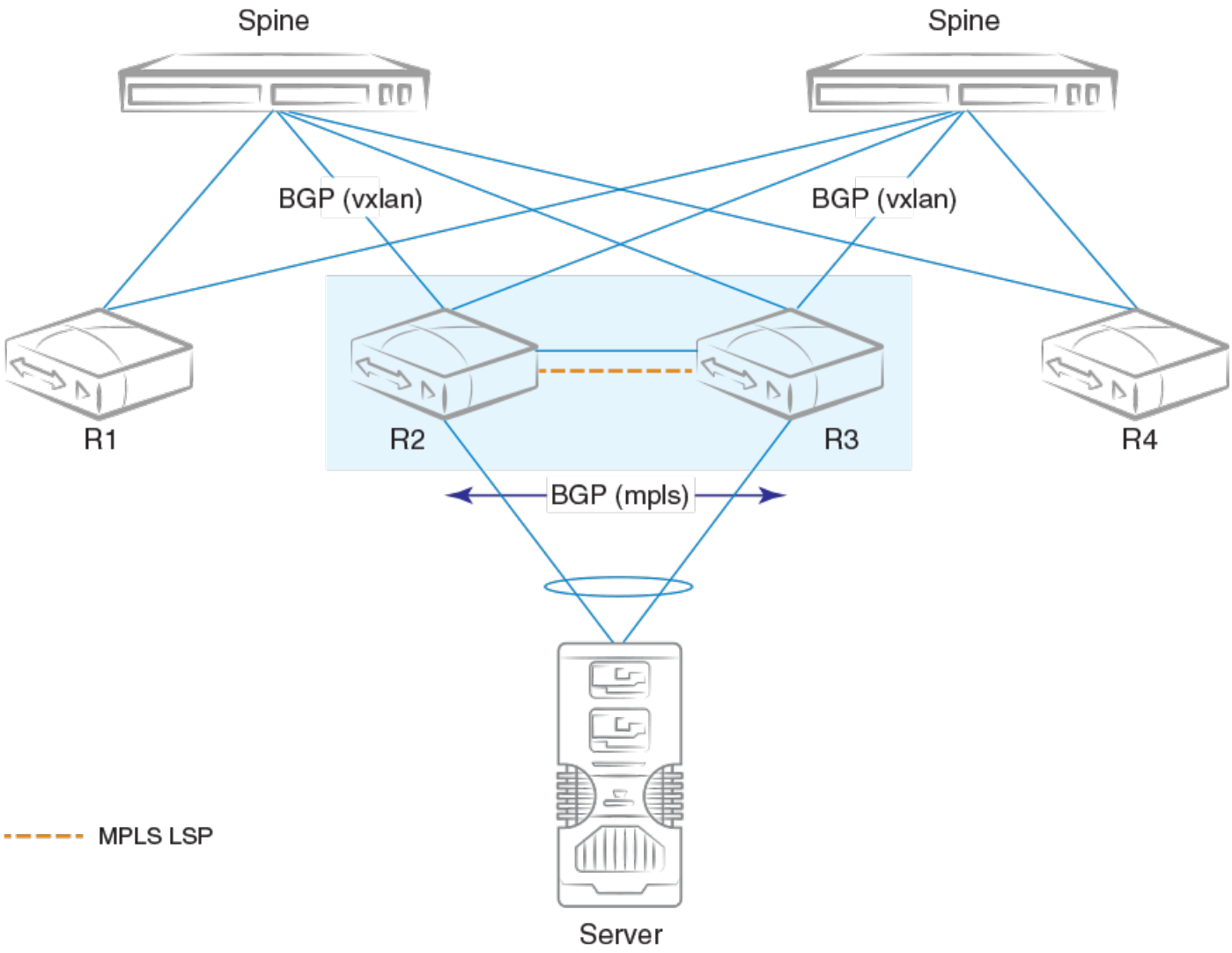
VLANs/BDs on static tunnels are not extended by BGP over inclusive-multicast routes received from remote nodes. They should be extended manually, as shown in the preceding configuration. Only when specific VLANs/BDs are extended does BGP download the corresponding EVPN routes pointing to the appropriate 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 25 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.

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 27 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 222. 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 28 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.

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 29 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.

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.

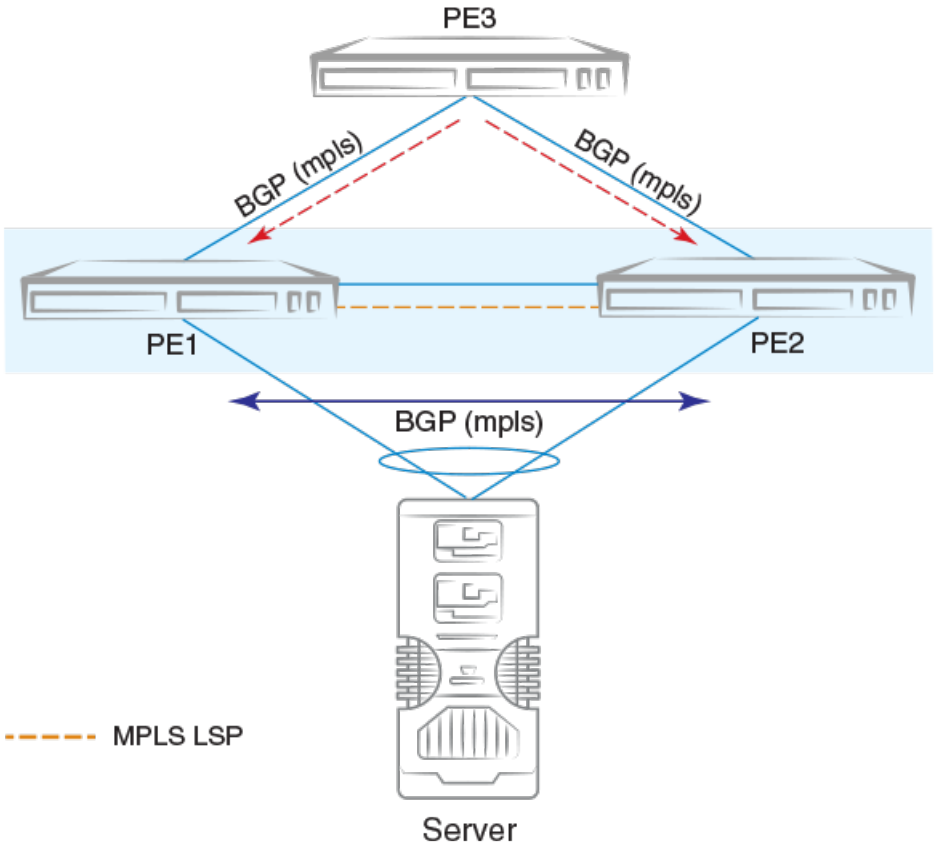
When BGP detects that only one source of the route (that is, all NLRIs are received from same next-hop) has remained and the route is dampened, the route is added to the timer list, which is processed after 5 minutes. When the timer expires, if the second source of the route is seen again, the route 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 26 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 30 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.

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.

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 31 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

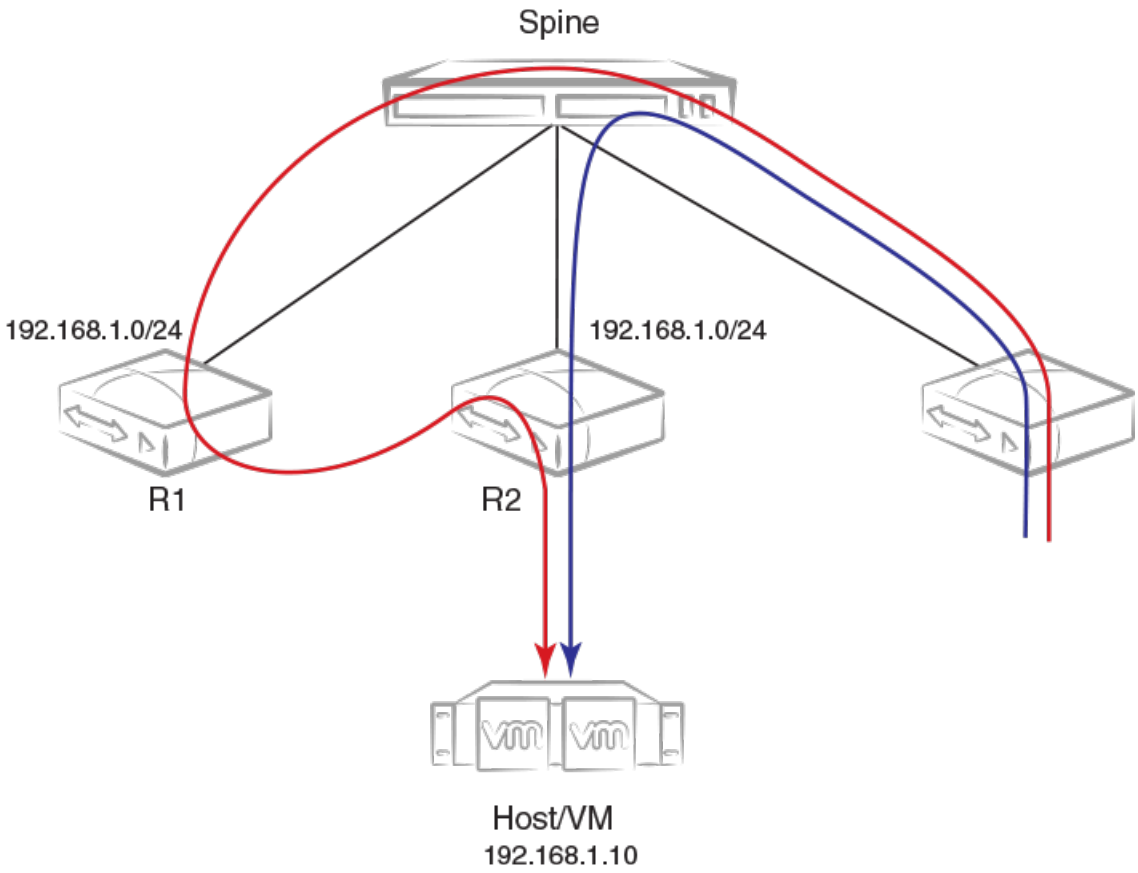
The route-map-based filtering of routes imported into IPVRF or exported into EVPN is supported. An import or export route-map can be applied under IPv4 or IPv6 address-family under a VRF configuration, as shown in the following example configuration.

```
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 27 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.

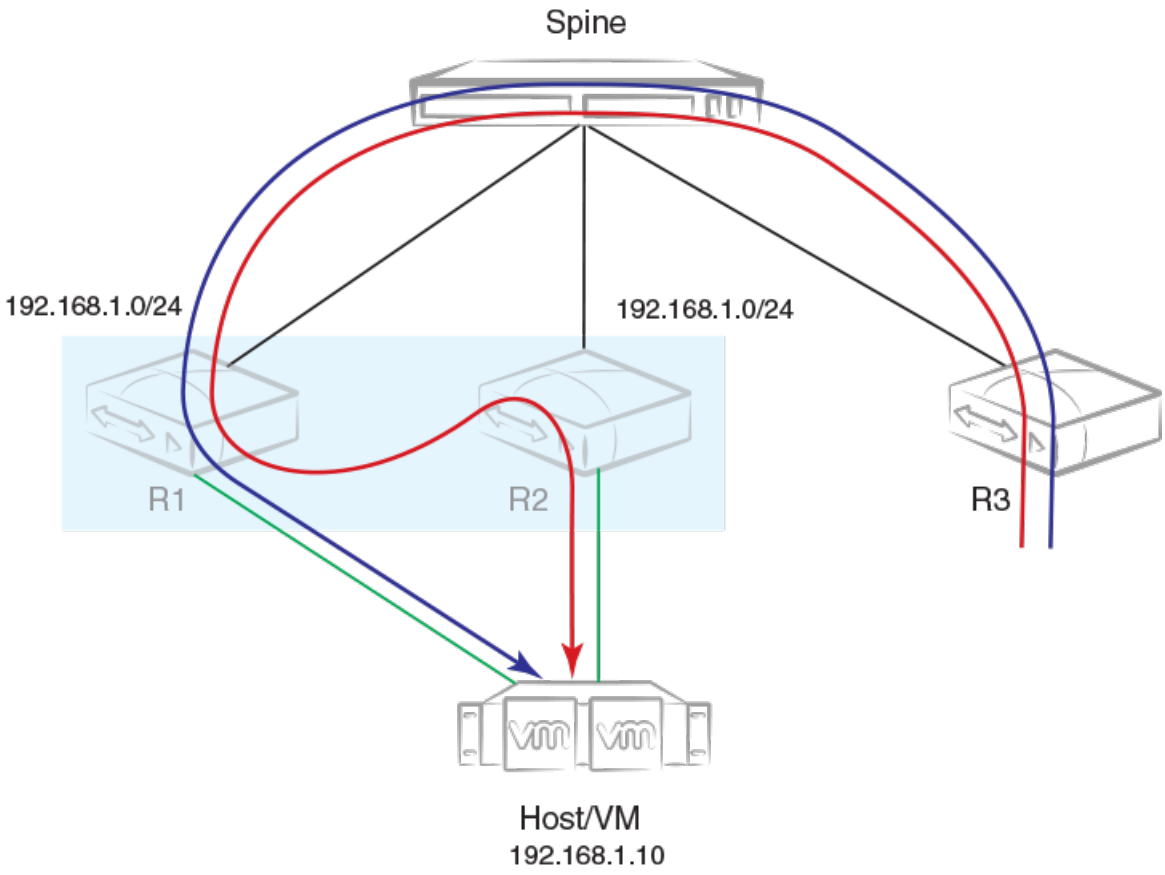
Avoiding traffic tromboning in an MCT cluster

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 28 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.

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.

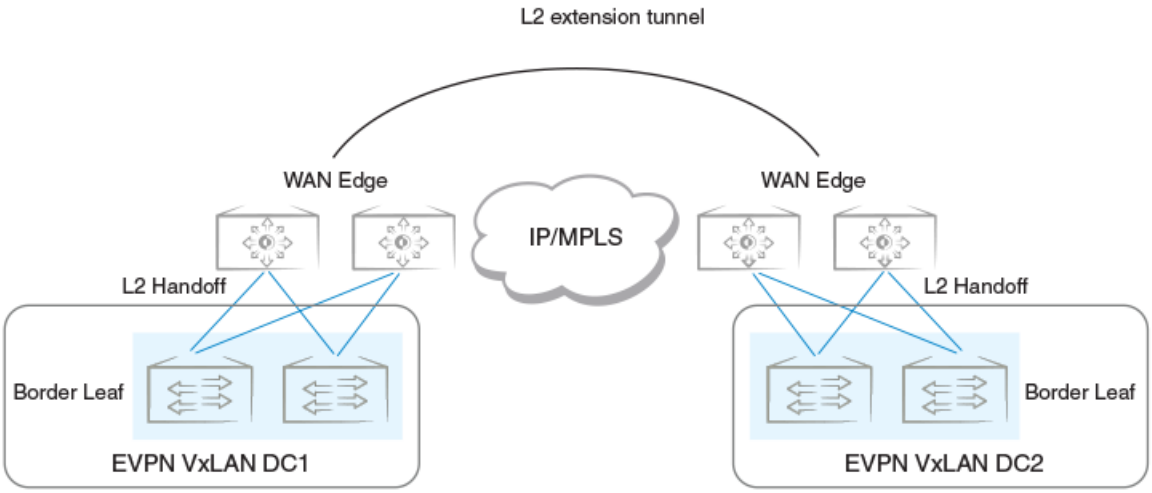
Layer 2 and Layer 3 control-plane extension

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.

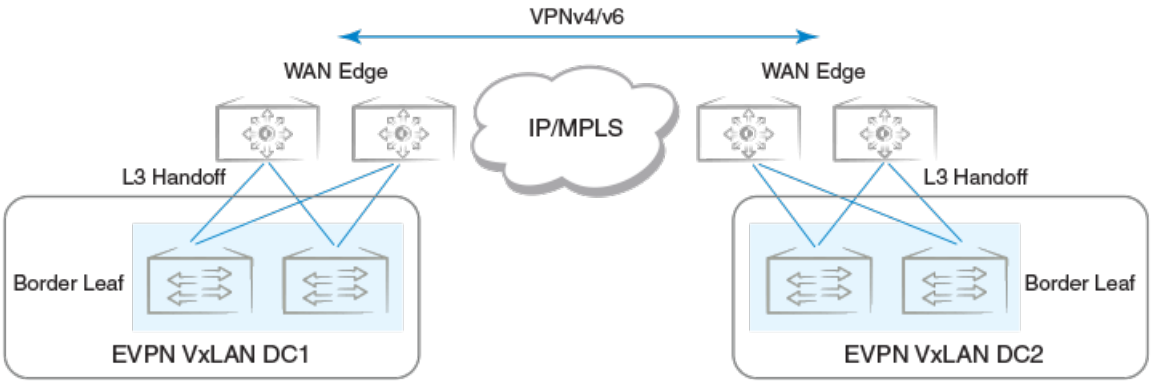
FIGURE 29 Layer 2 handoff-based DCI



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 30 Layer 3 handoff-based DCI



Static anycast gateway

Static anycast gateway functionality provides support for seamless VM mobility across the leaf switches in IP Fabric deployments.

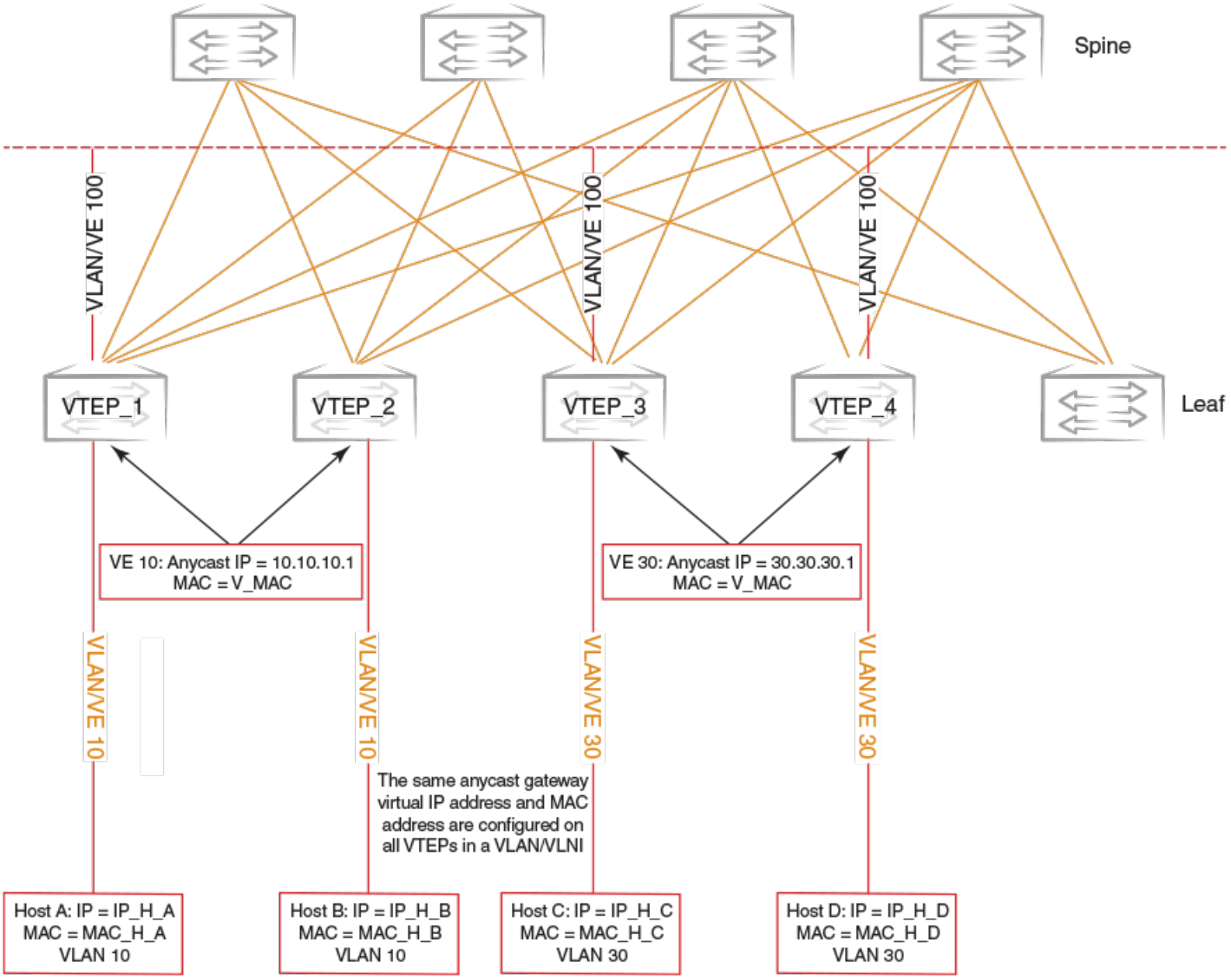
Hosts attached to the leaf switches are configured with a default gateway, which is typically the IP address of the leaf switch in the VLAN facing the host. If this host moves to another leaf switch, the host has to be reconfigured with the IP address of the new leaf switch to which it is now connected 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 31 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.

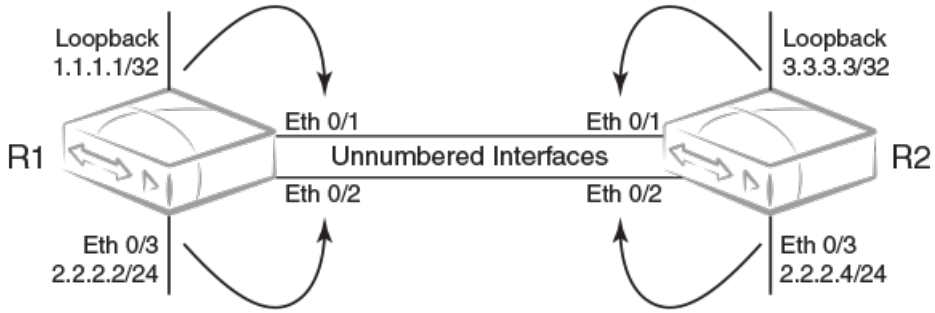
With large numbers of routers and redundant Layer 3 interfaces, a significant number of IP addresses are consumed just to configure the network itself. Using /31 masks reduces the consumption of addresses, but two IP addresses are still consumed per interface. Using unnumbered interfaces greatly reduces 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 32 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.

Generic Routing Encapsulation

- GRE tunnels....245
- Limitations and restrictions....246
- Configuring a Tunnel....246
- Binding a tunnel to a VE interface....247
- Enabling tunnel statistics....248
- Configuring GRE tunnel keepalive....248
- Configuring differentiated services codepoint....249
- Configuring MTU....249
- Displaying tunnel statistics....250
- Clearing tunnel statistics....251

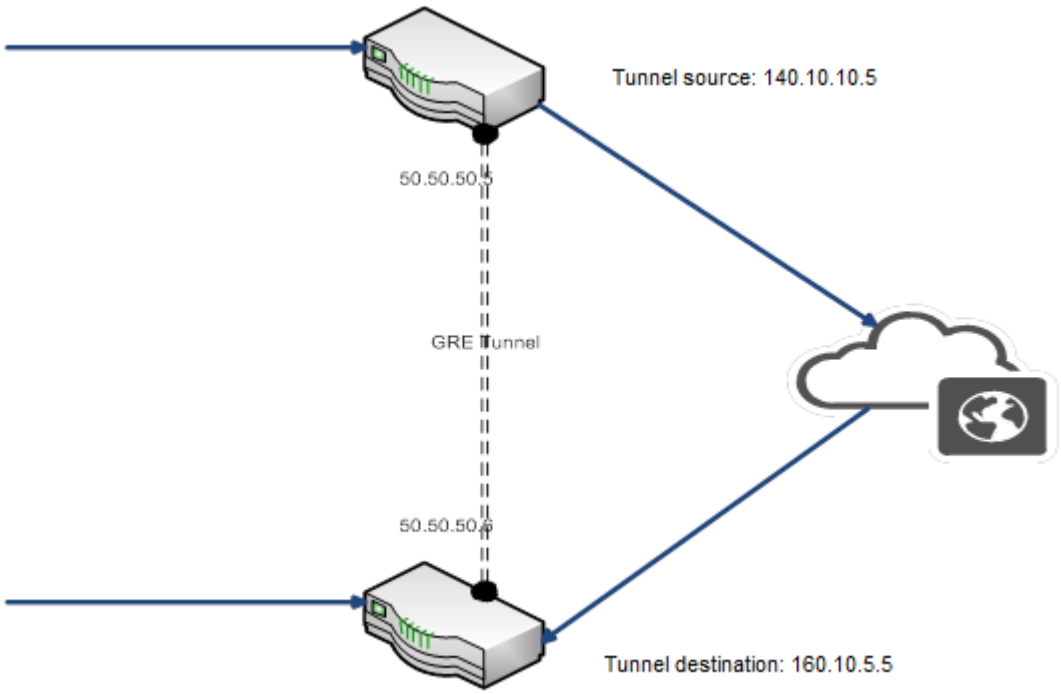
GRE tunnels

The Generic Routing Encapsulation (GRE) is a transport protocol that transmits other protocol packets. The GRE tunnels act as a virtual point-to-point connection identified by the tunnel source and the tunnel destination configuration. A tunnel interface can be associated with an IP interface. A GRE tunnel acts as a transport medium for the IP interface. In SLX-OS, a VE interface is associated with a GRE tunnel. The properties such as the IP address, routing protocol that are configured in the VE interface are be applied for the GRE tunnel.

Before configuring a GRE tunnel, a point-to-point tunnel configuration, GRE requires both the ends of the tunnel to be configured.

The diagram below illustrates the GRE tunnel.

FIGURE 33 GRE tunnel



The IP address of a tunnel is derived from the VE interface. After a VE is associated with a tunnel, all the configuration under the VE such as IP address, routing protocol configurations are applied to the GRE tunnel interface. GRE tunnels support pipe and uniform QoS mode. By default, the QoS mode is set to pipe mode for all the tunnels.

By default, statistics for a tunnel is disabled. You can enable statistics for each tunnel by using the **statistics** command. Note that traffic loss might occur when enabling or disabling statistics on a tunnel. GRE keep alive is an optional configuration. The minimum granularity of the GRE keep alive time is 1 second.

You can configure MTU for each tunnel interface. The MTU configured in a VE is used for tunnels. When the VE MTU is set to the default MTU 1500, the GRE tunnel MTU will be adjusted to 1476. If the VE MTU is set to any value other than the default value, the VE MTU is used as the GRE tunnel MTU.

Limitations and restrictions

The GRE feature has the following limitations and restrictions.

- The tunnel source and the destination addresses do not support VRF.
- Currently, only OSPF, ISIS, and BGP are supported on GRE tunnels.
- Supporting the QoS mutation configuration on the VE is not supported.
- The maximum number of GRE tunnels supported is 1024, the maximum number of supported in the default TCAM profile is 256, and the maximum number of tunnels supported in the VXLAN TCAM profile is 1024.
- The routing protocol used for resolving the GRE tunnel destination cannot be configured under the VE which is bound to the GRE tunnel.
- QoS mutation configuration on VE is not supported for VEs that are bound to GRE tunnels.
- IPv6 is not supported on GRE tunnels.

Configuring a Tunnel

To configure a GRE tunnel, perform the following tasks.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Use the **interface tunnel** command to enter interface configuration mode and configure a tunnel.

   ```
   device (config)# interface tunnel 5
   ```

   The range is from 1 through 1024.

3. Use the **mode gre ip** command to configure GRE encapsulation over IP as mode.

   ```
   device(config-intf-tunnel-5)# mode gre ip
   ```

4. Use the **source** command to configure the source IP address of the tunnel.

   ```
   device(config-intf-tunnel-5)# source 10.1.1.10
   ```

   The source IP address or interface that is configured will be used as the source of the GRE tunnel.

5. Use the **source** command to configure source interface of the tunnel.

```
device(config-intf-tunnel-5)# source ethernet ve 3
```

   Ethernet, loopback, and VE interfaces are supported. The maximum number of tunnel source supported is 16.

   NOTE
   When the physical/ve interface is specified as the source of the GRE tunnel, the lowest IP address of that interface is used as the tunnel source IP address. If the smallest IP address is removed from the interface, the next smallest IP address is used as the tunnel source.

6. Use the **destination** command to configure destination IP address.

```
device(config-intf-tunnel-5)# destination 10.1.1.11
```

7. Use **router- interface** command to configure a router interface for the tunnel.

```
device(config-intf-tunnel-5)# router-interface ve 4
```

   The tunnel source VE and the router interface VE cannot be the same.

8. Use the **no shutdown** command to configure the tunnel to remain up.

```
device(config-intf-tunnel-5)# no shutdown
```

The following example configures a GRE tunnel.

```
device# configure terminal
device (config)# interface ve 3
device(config-if-Ve-3)# ip address 50.50.50.5/24
device(config-if-Ve-3)# exit
device (config)# interface tunnel 5
device(config-intf-tunnel-5)# mode gre ip
device(config-intf-tunnel-5)# source 10.1.1.10
device(config-intf-tunnel-5)# source ve 4
device(config-intf-tunnel-5)# destination 10.1.1.11
device(config-intf-tunnel-5)# router-interface ve 3
device(config-intf-tunnel-5)# dscp-ttl-mode pipe
device(config-intf-tunnel-5)# no shutdown
```

Binding a tunnel to a VE interface

1. Enter global configuration mode.

```
device# configure terminal
```

2. Use the **interface** command to enter interface configuration mode and configure a VE interface.

```
device(config)# interface ve 3
```

   The range is from 1 through 4096.

3. Use the **ip address** command to assign an IP address to the VE interface.

```
device(config-if-Ve-3)# ip address 10.1.1.1
```

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

```
device(config-if-Ve-3)# exit
```

5. Use the **interface** command to enter interface configuration mode and configure a tunnel.

```
device(config)# interface tunnel 5
```

   The range is from 1 through 1024.

6. Use the **router-interface** command to configure a router interface for the tunnel.

```
device(config-intf-tunnel-5)# router-interface ve 3
```

This example shows how to bind a tunnel to a VE interface.

```
device# configure terminal
device (config)# interface ve 3
device(config-if-Ve-3)# ip address 10.1.1.1
device(config-if-Ve-3)# exit
device (config)# interface tunnel 5
device(config-intf-tunnel-5)# router-interface ve 3
```

Enabling tunnel statistics

1. Enter global configuration mode.

```
device# configure terminal
```

2. Use the **interface** command to enter interface configuration mode and configure a tunnel.

```
device(config)# interface tunnel 5
```

   The range is from 1 through 1024.

3. Use the **statistics** command to enable statistics on the tunnel.

```
device(config-intf-tunnel-5)# statistics
```

The following example shows how to enable tunnel statistics.

NOTE
GRE tunnel statistics only counts traffic entering the GRE tunnel. Traffic egressing the GRE tunnel is not counted.

```
device# configure terminal
device (config)# interface tunnel 5
device(config-intf-tunnel-5)# statistics
```

Configuring GRE tunnel keepalive

1. Enter global configuration mode.

```
device# configure terminal
```

2. Use the **interface** command to enter interface configuration mode and configure a tunnel.

```
device(config)# interface tunnel 5
```

   The range is from 1 thorough 1024.

3. Use the **keepalive** command to configure tunnel keep alive.

```
device(config-intf-tunnel-5)# keepalive time-interval 30 retry-count 2
```

   The range for time interval is from 1 to 32767. The range for retries is from 1 thorough 255.

The following example shows how to configure GRE tunnel keep alive.

```
device# configure terminal
device (config)# interface tunnel 5
keepalive time-interval 30 retry-count 2
```

Configuring differentiated services codepoint

1. Enter global configuration mode.

```
device# configure terminal
```

2. Use the **interface** command to enter interface configuration mode and configure a tunnel.

```
device(config)# interface tunnel 5
```

   The range is from 1 thorough 1024.

3. Use the **dscp** command configure tunnel differentiated services codepoint (DSCP).

```
device(config-intf-tunnel-5)# dscp 10
```

   The range is from 0 thorough 63.

The following example shows how to configure tunnel DSCP.

```
device# configure terminal
device (config)# interface tunnel 5
device(config-intf-tunnel-5)# dscp 10
```

Configuring MTU

1. Enter global configuration mode.

```
device# configure terminal
```

2. Use the **interface** command to enter interface configuration mode and configure a VE interface.

```
device (config)# interface ve 5
```

   The range is from 1 through 4096.

3. Use the **ip mtu** command to configure MTU on the interface.

```
device(config-intf-ve-6)# ip mtu 9011
```

   The range is from 1300 through 9194. The default MTU is 1476.

The following example shows how to configure MTU for an interface.

> **NOTE**
> Only the default MTU of 1476 is supported for GRE tunnels. IP packets up to 1476 bytes can be transmitted on the GRE tunnels. Packet higher than 1476 can go unfragmented and can cause the dropping of packets in the next nodes. Extreme recommends that you set the MTU of the traffic source interface to 1476, so that no fragmentation is needed.

```
device# configure terminal
device (config)# interface ve 5
device(config-intf-ve-6)# ip mtu 9011
```

Displaying tunnel statistics

1. From user EXEC mode, use the **show tunnel statistics** command to display statistics of all tunnels.

   ```
   device# show tunnel statistics
   ```

2. Enter the tunnel ID to display statistics of a specific tunnel.

   ```
   device# show tunnel statistics 7
   ```

The following example shows how to display statistics of all tunnels.

```
device# show tunnel statistics
```

The following example shows how to display statistics of the specified tunnel ID.

```
device# show tunnel statistics mode gre
Tnl ID   RX packets      TX packets      RX bytes        TX bytes
======== =============== =============== =============== ================
5        0               0               (NA)            0
```

The following example shows how to display tunnel statistics mode.

```
device# show tunnel statistics mode gre
Tnl ID   RX packets      TX packets      RX bytes        TX bytes
======== =============== =============== =============== ================
10       0               10               (NA)            640
11       0               20               (NA)            1280
12       0               50               (NA)            22000
```

The following example shows how to display information of the specified tunnel.

```
device# show tunnel 10
Tunnel 10, mode GRE
Ifindex 0x7c40000a, Admin state up, Oper state up
Source IP 14.101.0.4, Vrf default-vrf
Destination IP 15.10.0.3
Tunnel IP Interface : Ve 501 up
Tunnel TTL 255      Tunnel DSCP 0
Tunnel QosMode PIPE
Keepalive Interval 10000  RetryCount 3 TimeRemaining 27861 msecs
GRE Keep Alive : RX 62      TX 62

Active next hops:
    IP: 13.10.0.3, Vrf: default-vrf
    Egress L3 port: Ve 10, Outer SMAC: 609c.9f0d.4a14
    Outer DMAC: 001b.ed9f.1700
    Egress L2 Port: Unknown, Outer ctag: 0, stag:0, Egress mode: Local
    BUM forwarder: no
```

Clearing tunnel statistics

1. From Privileged EXEC mode, use the **clear tunnel statistics** command to clear statistics of all tunnels.

```
device# clear tunnel statistics
```

2. Enter the tunnel ID to clear statistics of a specific tunnel.

```
device# clear tunnel statistics 7
```

The following example shows how to clear statistics of all tunnels.

```
device# clear tunnel statistics
```

The following example shows how to clear statistics of the specified tunnel.

```
device# clear tunnel statistics 7
```

IS-IS (IPv4)

- IS-IS overview....254
- Relationship to the IP route table....254
- IS-IS CLI levels....258
- Enabling IS-IS globally....258
- Configuring the IS-IS IPv4 unicast address family....259
- Overload bit....259
- Authentication....260
- Changing the IS-IS level globally....261
- Logging adjacency changes....262
- Complete Sequence Numbers PDU interval....262
- Changing the maximum LSP lifetime....263
- Changing the LSP refresh interval....263
- Changing the LSP generation interval....264
- Changing the LSP interval and retransmit interval....264
- Disabling IS-IS name mapping capability....265
- Logging invalid LSP packets received....265
- Changing the SPF timer....265
- Configuring the IS-IS flooding mechanism....266
- Configuring IS-IS PSPF exponential back-off....267
- Hello padding....267
- Partial SPF optimizations....268
- Incremental SPF optimizations....269
- IS-IS incremental shortcut LSP SPF optimization....269
- Maximum number of load sharing paths....270
- Default route advertisement....271
- IS-IS administrative distance....272
- Configuring summary addresses....273
- IPv4 IS-IS route redistribution....274
- Default redistribution metric....275
- Configuring the default link metric value globally....276
- IS-IS metric styles....276
- IS-IS non-stop routing....277
- Limitations of IS-IS non-stop routing....278
- Enabling IS-IS for an Interface....279
- Configuring authentication on an IS-IS interface....279
- Disabling hello padding for an IS-IS interface....280
- Changing the IS-IS level on an IS-IS interface....280
- Changing the hello multiplier for an IS-IS interface....281
- Changing the hello interval for an IS-IS interface....281
- IS-IS point-to-point over Ethernet....282
- Displaying IS-IS statistics....284

IS-IS overview

The Intermediate System to Intermediate System (IS-IS) protocol is a link-state Interior Gateway Protocol (IGP) that is based on the International Standard for Organization/ International Electrotechnical Commission (ISO/IEC) Open Systems Internet Networking model (OSI). In IS-IS, an intermediate system (router) is designated as either a Level 1 or Level 2 device. A Level 1 router routes traffic only within the area in which the router resides. A Level 2 router routes traffic between areas within a routing domain.

The implementation of IS-IS is based on the following specifications and draft specifications:

- ISO/IEC 10589 - "Information Technology - Telecommunication and information exchange between systems - Intermediate system to Intermediate system intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connection less-mode Network Service (ISO 8473)", 1992
- ISO/IEC 8473 - "Information processing systems - Data Communications - Protocols for providing the connectionless-mode network service", 1988
- ISO/IEC 9542 - "Information Technology - Telecommunication and information exchange between systems - End system to Intermediate system intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connection less-mode Network Service (ISO 8473)", 1988
- RFC 1195 - "Use of OSI IS-IS for Routing in TCP/IP and Dual Environments", 1990.
- RFC 2763 - "Dynamic Host Name Exchange Mechanism for IS-IS", 2000.
- RFC 2966 - "Domain-wide Prefix Distribution with Two-Level IS-IS", 2000
- RFC 3373 - "Three-Way Handshake for Intermediate System to Intermediate System (IS-IS) Point-to-Point Adjacencies", 2002
- Portions of the Internet Draft "IS-IS extensions for Traffic Engineering" draft-ieff-isis-traffic-02.txt (dated 2000). that describe the Extended IP reachability type-length-value (TLV type 135) and the extended Intermediate System (IS) reachability TLV (TLV type 22). These portions provide support for the wide metric version of IS-IS. No other portion is supported on Extreme's implementation of IS-IS.

  NOTE
  The Extreme device does not support routing of Connectionless-Mode Network Protocol (CLNP) packets. The Extreme device uses IS-IS for TCP/IP only.

Relationship to the IP route table

The IS-IS routes are calculated and first placed in the IS-IS route table. The routes are then transferred to the IP route table.

The best IS-IS path for a given destination is sent to the IP route table for comparison to the best paths from other protocols to the same destination. The CPU selects the path with the lowest administrative distance and places that path in the IP route table:

- If the path provided by IS-IS has the lowest administrative distance, then the CPU places that IS-IS path in the IP route table.
- If a path to the same destination supplied by another protocol has a lower administrative distance, the CPU installs the other protocol's path in the IP route table instead.

The **administrative distance** is a protocol-independent value from 1 - 255. Each path sent to the CPU, regardless of the source of the path (IS-IS, OSPF, static IP route, and so on) has an administrative distance.

Each route source has a default administrative distance. The default administrative distance for IS-IS is 115.

You can change the administrative distance for IS-IS and other routes sources.

Intermediate systems and end systems

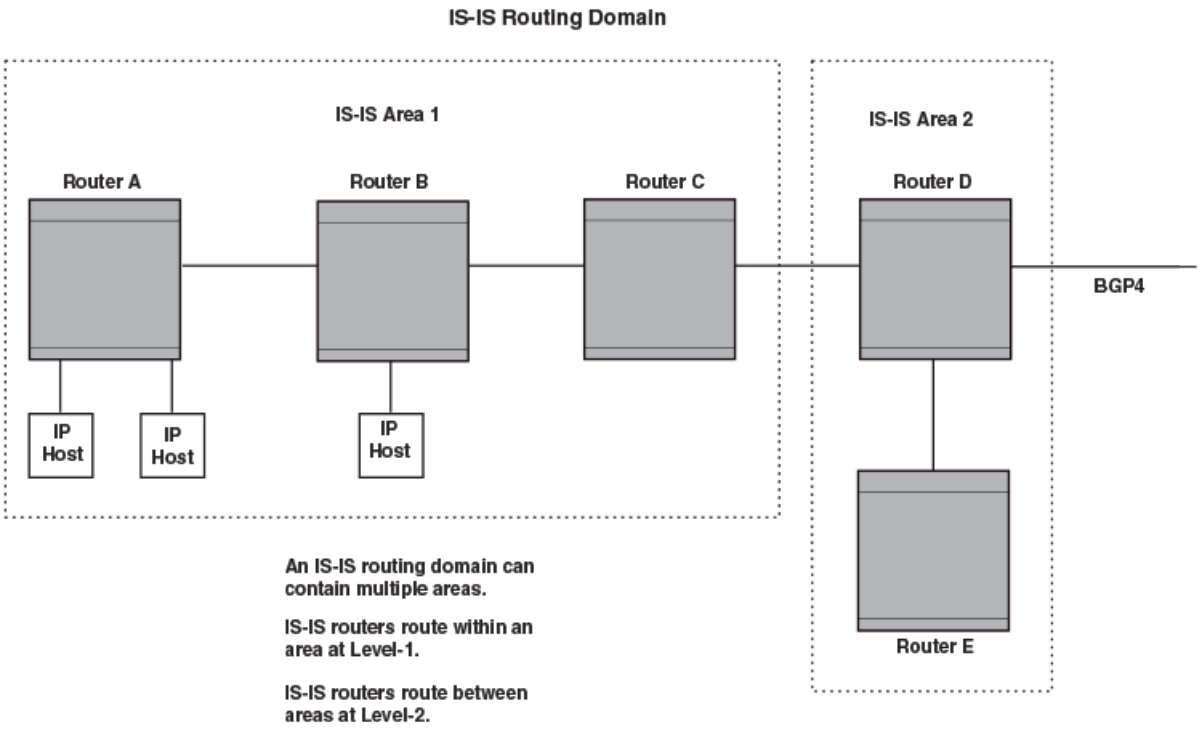
IS-IS uses the following categories to describe devices within an IS-IS routing domain (similar to an OSPF Autonomous System):

- Intermediate System (IS) - A device capable of forwarding packets from one device to another within the domain. In Internet Protocol (IP) terminology, an IS is a router.
- End System (ES) - A device capable of generating or receiving packets within the domain. In IP terminology, an ES is an end node or IP host.

When you configure IS-IS on a device, the device is an IS.

Figure 34 shows an example of an IS-IS network.

FIGURE 34 An IS-IS network contains Intermediate Systems (ISs) and host systems



NOTE

Since the implementation of IS-IS does not route OSI traffic but instead routes IP traffic, IP hosts are shown instead of ESs.

The other basic IS-IS concepts illustrated in this figure are explained in the following sections.

Domain and areas

IS-IS is an IGP, and thus applies only to routes within a single routing domain. However, you can configure multiple areas within a domain. A device can be a member of one area for each Network Entity Title (NET) you configure on the device. The NET contains the area ID for the area the NET is in.

In Intermediate systems and end systems on page 255, Routers A, B, and C are in area 1. Routers D and E are in area 2. All the routers are in the same domain.

Level-1 routing and Level-2 routing

You can configure an IS-IS router to perform one or both of the following levels of IS-IS routing:

> NOTE
> The ISO/IEC specifications use the spelling "routeing", but this document uses the spelling "routing" to remain consistent with other Extreme documentation.

- Level-1 - A Level-1 router routes traffic only within the area the router is in. To forward traffic to another area, the Level-1 router sends the traffic to its nearest Level-2 router.
- Level-2 - A Level-2 router routes traffic between areas within a domain.

In Intermediate systems and end systems on page 255, Routers A and B are Level-1s only. Routers C and D are Level-1 and Level-2 ISs. Router E is a Level-1 IS only.

Neighbors and adjacencies

A device configured for IS-IS forms an adjacency with each of the IS-IS devices to which it is directly connected. An adjacency is a two-way direct link (a link without router hops) over which the two devices can exchange IS-IS routes and other protocol-related information. The link is sometimes called a “circuit”. The devices with which the device forms adjacencies are its neighbors , which are other ISs.

In Intermediate systems and end systems on page 255, Router A has an IS-IS adjacency with Router B. Likewise, Router B has an IS-IS adjacency with Router A and Router C.

Designated IS

A Designated IS is an IS-IS router that is responsible for gathering and distributing link state information to other Level 1 or Level 2 ISs within the same broadcast network (LAN). The Level 1 and Level 2 Designated ISs within a broadcast network are independent, although the same device can be a Level 1 Designated IS and a Level 2 Designated IS at the same time.

The Designated IS is elected based on the priority of each IS in the broadcast network. When an IS becomes operational, it sends a Level 1 or Level 2 Hello PDU to advertise itself to other ISs. If the IS is configured to be both a Level 1 and a Level 2 IS, the IS sends a separate advertisement for each level:

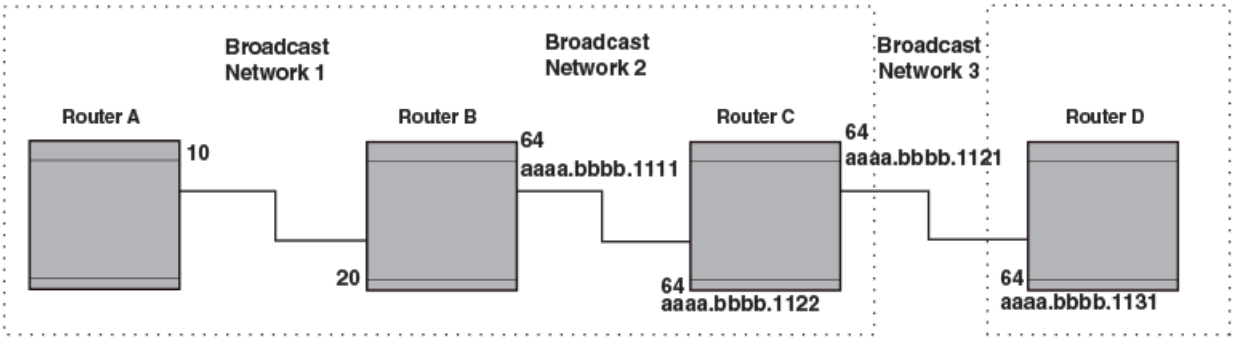
- The Level 1 IS that has the highest priority becomes the Level 1 Designated IS for the broadcast network.
- The Level 2 IS that has the highest priority becomes the Level 2 Designated IS for the broadcast network.

If the Designated IS becomes unavailable, for example in the case of a reboot, the IS with the next highest priority becomes the new IS. If two or more ISs have the highest priority, the IS with the highest MAC address becomes the Designated IS.

The priority is an interface parameter. Each interface that is enabled for IS-IS can have a different priority.

The figure below shows an example of the results of Designated IS elections. For simplicity, this example shows four of the five routers in Intermediate systems and end systems on page 255, with the same domain and areas.

FIGURE 35 Each broadcast network has a Level-1 Designated IS and a Level-2 Designated IS



Designated IS election has the following results in this network topology:

- Router B is the Level 1 Designated IS for broadcast network 1
- Router C is the Level 1 Designated IS for broadcast network 2
- Router D is the Level 2 Designated IS for broadcast network 3

In this example, the IS-IS priorities for the IS-IS interfaces in broadcast network 1 have been changed by an administrator. The priorities for the interfaces in the other broadcast networks are still set to the default (64). When there is a tie, IS-IS selects the interface with the highest MAC address.

Broadcast pseudonode

In a broadcast network, the Designated IS maintains and distributes link state information to other ISs by maintaining a pseudonode . A pseudonode is a logical host representing all the Level 1 or Level 2 links among the ISs in a broadcast network. Level 1 and Level 2 have separate pseudonodes, although the same device can be the pseudonode for Level 1 and Level 2.

Route calculation and selection

The Designated IS uses a Shortest Path First (SPF) algorithm to calculate paths to destination ISs and ESs. The SPF algorithm uses Link State PDUs (LSPDUs) received from other ISs as input, and creates the paths as output.

After calculating the paths, the Designated IS then selects the best paths and places them in the IS-IS route table. The Designated IS uses the following process to select the best paths.

1. Prefer the Level 1 path over the Level 2 path.
2. If there is no Level 1 path, prefer the internal Level 2 path over the external Level 2 path.
3. If there is still more than one path, prefer the path with the lowest metric.
4. If there is more than one path with the lowest metric, load share among the paths.

   After selecting the best path to a destination, the software places the path in the IS-IS route table.

Three-way handshake for point-to-point adjacencies

Three-Way Handshake for Point-to-Point adjacencies provides three-way handshake mechanisms on point-to-point interfaces for the following benefits:

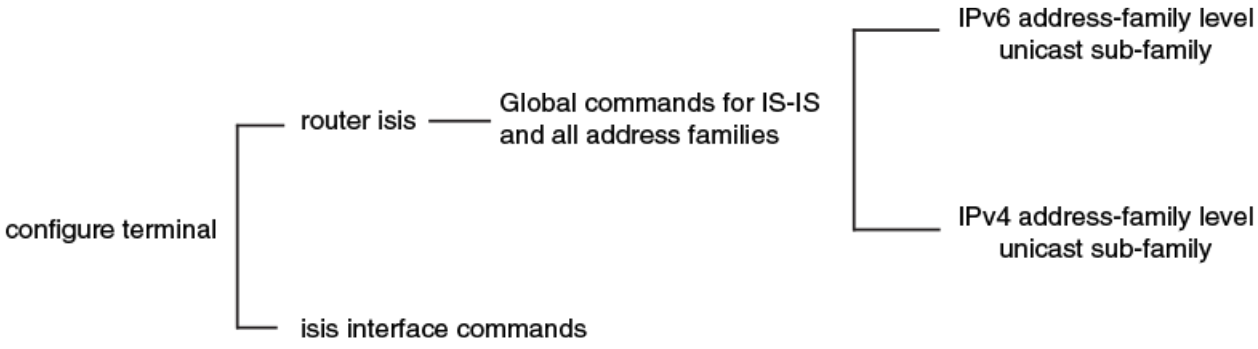
- Identifies neighbor restarts within the holding time period

- Identifies uni-directional link failures and stops forming of an adjacency with a peer where such link failures occur.

IS-IS CLI levels

The CLI includes various levels of commands for IS-IS. The figure below illustrates these levels.

FIGURE 36 IS-IS CLI levels



The following IS-IS CLI configuration modes are available

- ISIS router configuration mode: A global level for the configuration of the IS-IS protocol. At this level, all IS-IS configurations apply to IPv4 and IPv6.
- ISIS address-family IPv4 unicast configuration mode: Commands entered in ISIS address-family IPv4 unicast configuration mode apply only to IS-IS IPv4 configurations.
- ISIS address-family IPv6 unicast configuration mode: Commands entered in ISIS address-family IPv6 unicast configuration mode apply only to IS-IS IPv6 configurations.
- Interface subtype configuration mode: Various IS-IS options are configured on a specified interface.

Refer to the relevant Command Reference for more information on each of these configuration modes and for information on all of the commands accessed in these configuration modes.

Each address family configuration level allows you to access commands that apply to that particular address family only. To enable a feature in a particular address family, you must specify any associated commands for that feature in that particular address family. You cannot expect the feature, which you may have configured in the IPv4 IS-IS unicast address family, to work in the IPv6 IS-IS unicast address family unless it is explicitly configured in the IPv6 IS-IS unicast address family.

Enabling IS-IS globally

When IS-IS is enabled on a device, the device enters IS-IS router configuration mode. Several commands can then be accessed that allow the configuration of IS-IS.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
ISIS: Please configure NET!
```

3. Enter the **net** command and specify a NSAP address to configure a NET for IS-IS.

```
device(config-isis-router)# net 49.2211.0000.00bb.cccc.00
```

The following example enables IS-IS on a device.

```
device# configure terminal
device(config)# router isis
ISIS: Please configure NET!
device(config-isis-router)# net 49.2211.0000.00bb.cccc.00
```

Configuring the IS-IS IPv4 unicast address family

The IS-IS IPv4 unicast address family allows you to configure IPv4 IS-IS unicast settings that are separate and distinct from IPv6 IS-IS unicast settings. The following task enables ISIS IPv4 address family unicast configuration mode where a variety of IS-IS features can be configured.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

```
device(config-isis-router)# address-family ipv4 unicast
```

The following example enables ISIS address-family IPv4 unicast configuration mode.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)#
```

Overload bit

If an IS's resources are overloaded and are preventing the IS from properly performing IS-IS routing, the IS can inform other ISs of this condition by setting the overload bit in LSPDUs sent to other ISs from 0 (off) to 1 (on). Thus, when an IS is overloaded, other ISs will not use the overloaded IS to forward traffic.

An IS can be in the overload state for Level 1 or Level 2 as follows:

- If an IS is in the overload state for Level 1, other Level 1 ISs stop using the overloaded IS to forward Level 1 traffic. However, the IS can still forward Level 2 traffic, if applicable.
- If an IS is in the overload state for Level 2, other Level 2 ISs stop using the overloaded IS to forward Level 2 traffic. However, the IS can still forward Level 1 traffic, if applicable.
- If an IS is in the overload state for both levels, the IS cannot forward traffic at either level.

By default, the device automatically sets the overload bit to 1 (on) in its LSPDUs to other ISs if an overload condition occurs.

The overload bit can also be set to administratively shut down IS-IS without disabling the protocol. Setting the overload bit on is useful when configuration changes are needed but you do not want to remove the device from the network. A device can also be configured to set the overload bit on for a specific number of seconds during startup, to allow IS-IS to become fully active before the device begins IS-IS routing.

Setting the overload bit

The overload bit can be configured so that when an IS is overloaded, other ISs will not use the overloaded IS to forward traffic. The following task specifies that the overload bit is set upon system startup and remains set until BGP has converged and specifies that the device that 240 seconds is the maximum time that IS-IS will wait for BGP convergence to complete.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **set-overload-bit** command with the **on-startup** and **wait-for-bgp** parameters, specifying a value, to configure the device to set the overload bit upon system startup. The overload bit remains set until BGP has converged. 86,400 seconds is the maximum time that IS-IS waits for BGP convergence to complete.

```
device(config-isis-router)# set-overload-bit on-startup wait-for-bgp 86400
```

The following example configures the overload bit on system startup until BGP has converged. The maximum time that the device waits for BGP convergence to complete is 86400 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# set-overload-bit on-startup wait-for-bgp 86400
```

Authentication

By default, a device does not authenticate packets sent to or received from an end system (ES) or other intermediate system (IS).

An authentication password can be configured using the Hashed Message Authentication codes - Message Digest 5 (HMAC-MD5) algorithm, in conformance with RFC 3567 - Intermediate System to Intermediate System (IS-IS) Cryptographic Authentication.

IS-IS authentication checking is enabled by default. When transitioning from one authentication mode to another, changing the authentication mode can cause packets to drop because only some of the routers have been reconfigured. During such a transition, it can be useful to disable IS-IS authentication checking temporarily until all devices are reconfigured and the network is stable.

Authentication can be configured globally or for a specified interface.

To configure IS-IS authentication globally you must perform the following tasks:

- Configure IS-IS Authentication Mode
- Configure IS-IS Authentication Key
- Disable IS-IS Authentication Check (optional)

To configure IS-IS authentication on a specified interface, you must perform the following tasks:

- Configure IS-IS Interface Authentication Mode for the specified interface
- Configure IS-IS Authentication Key for the specified interface

- Disable IS-IS Authentication Check for the specified interface (optional)

Configuring authentication

A device can be configured to authenticate packets sent to or received from an end system (ES) or other intermediate system (IS). The following task configures IS-IS authentication mode, an IS-IS authentication key and temporarily disables IS-IS authentication checking for Level 1 packets.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **auth-mode** command with the **md5** and **level-1** parameters to configure MD5 authentication for Level 1 packets.

```
device(config-isis-router)# auth-mode MD5 level-1
```

4. Enter the **auth-check disable** command with the **level-1** parameter to temporarily disable authentication checking globally for Level 1 packets.

```
device(config-isis-router)# auth-check level-1 disable
```

5. Enter the **auth-key** command with the **level-1** parameters, specifying an authentication password to configure an authentication key globally for Level 1 packets.

```
device(config-isis-router)# auth-key level-1 mysecurekey
```

The following example configures IS-IS authentication mode, an IS-IS authentication key and temporarily disables IS-IS authentication checking for Level 1 packets.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# auth-mode MD5 level-1
device(config-isis-router)# auth-check level-1 disable
device(config-isis-router)# auth-key level-1 mysecurekey
```

Changing the IS-IS level globally

By default, a device can operate as both a Level 1 and Level 2 router. This task globally changes the level supported from Level 1 and Level 2 to Level 1 only.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **is-type** command with the **level-1** parameter to globally change the IS-IS level supported from Level-1 and Level-2 to Level-1 only.

```
device(config-isis-router)# is-type level-1
```

The following example globally changes the IS-IS level supported from Level-1 and Level-2 to Level-1 only.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# is-type level-1
```

Logging adjacency changes

You can configure a device to log changes in the status of an adjacency with another IS. The following task enables the logging of adjacency changes.

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **log adjacency** command to enable the logging of adjacency changes.

   ```
   device(config-isis-router)# log adjacency
   ```

The following example enables the logging of adjacency changes.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# log adjacency
```

Complete Sequence Numbers PDU interval

A Complete Sequence Numbers PDU (CSNP) is a complete list of the LSPs in the Designated IS' link state database. The CSNP contains a list of all the LSPs in the database, as well as other information that helps IS neighbors determine whether their LSP databases are in sync with one another. The Designated IS sends CSNPs to the broadcast interface. Level 1 and Level 2 each have their own Designated IS.

A Partial Sequence Numbers PDU (PSNP) is a partial list of LSPs. ISs other than the Designated IS (that is, the non-Designated ISs) send PSNPs to the broadcast interface. The CSNP interval specifies how often the Designated IS sends a CSNP to the broadcast interface. Likewise, the PSNP interval specifies how often other ISs (non-Designated ISs) send a PSNP to the broadcast interface.

Configuring the CSNP interval

By default the Complete Sequence Numbers PDU (CSNP) interval is 10 seconds. The following task configures CSNP interval of 25 seconds for Level 1 and Level 2 packets.

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **csnp-interval** command and specify a value to globally configure the CSNP interval.

```
device(config-isis-router)# csnp-interval 25
```

The following example configures a CSNP interval of 25 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# csnp-interval 25
```

Changing the maximum LSP lifetime

This task changes the maximum number of seconds an unrefreshed LSP remains in a device's LSP database from the default of 1200 seconds (20 minutes) to 2400 seconds (40 minutes).

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **max-lsp-lifetime** command and specify a value to globally configure the maximum LSP lifetime.

```
device(config-isis-router)# max-lsp-lifetime 2400
```

The following example changes the maximum LSP lifetime to 2400 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# max-lsp-lifetime 2400
```

Changing the LSP refresh interval

This task changes the maximum number of seconds a device waits between sending updated LSPs to its IS-IS neighbors from the default of 900 seconds to 1800 seconds.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **lsp-refresh-interval** command and specify a value to change the LSP refresh interval.

```
device(config-isis-router)# lsp-refresh-interval 1800
```

The following example changes the LSP refresh interval to 1800 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# lsp-refresh-interval 1800
```

Changing the LSP generation interval

This task changes the minimum number of seconds the device waits between sending updated LSPs to its IS-IS neighbors from the default of 10 seconds to 60 seconds (one minute).

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **lsp-gen-interval** command and specify a value to change the LSP generation interval.

```
device(config-isis-router)# lsp-gen-interval 60
```

The following example changes the LSP generation interval to 60 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# lsp-gen-interval 60
```

Changing the LSP interval and retransmit interval

The LSP interval is the rate of transmission, in milliseconds, of the LSPs. The retransmit interval is the time, in seconds, the device waits before it retransmits LSPs. This task changes the LSP interval to 45 milliseconds and changes the retransmission interval to 10 milliseconds.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **lsp-interval 45** command and specify a value to change the LSP interval from the default of 33 milliseconds.

```
device(config-isis-router)# lsp-interval 42
```

4. Enter the **retransmit-interval** command and specify a value to change the retransmission interval from the default of 5 seconds.

```
device(config-isis-router)# retransmit-interval 10
```

The following example changes the LSP interval to 45 milliseconds and changes the retransmission interval to 10 milliseconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# lsp-interval 42
device(config-isis-router)# retransmit-interval 10
```

Disabling IS-IS name mapping capability

The implementation of IS-IS supports RFC 2763, which describes a mechanism for mapping IS-IS system IDs to the hostnames of the devices with those IDs. The following task disables IS-IS name mapping.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **hostname disable** command to disable IS-IS name mapping.

```
device(config-isis-router)# hostname disable
```

The following example globally disables IS-IS name mapping.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# hostname disable
```

Logging invalid LSP packets received

You can configure a device to provide logging of invalid LSP packets. The following task enables the logging of invalid LSP packets.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **log invalid-lsp-packets** command to enable the logging of invalid LSP packets.

```
device(config-isis-router)# log invalid-lsp-packets
```

The following example enables the logging of invalid LSP packets.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# log invalid-lsp-packets
```

Changing the SPF timer

Every IS maintains a Shortest Path First (SPF) tree, which is a representation of the states of each of the IS's links to ESs and other ISs. If the IS is both a Level-1 and Level-2 IS, it maintains separate SPF trees for each level. To ensure that the SPF tree remains current, the IS updates the tree at regular intervals following a change in network topology or the link state database. This task changes the maximum

interval in seconds between SPF recalculations, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **spf-interval** command with the **level-1** parameter, specifying values for the *max-wait* , *initial-wait* , and *second-wait* parameters, to set the maximum interval in seconds between SPF recalculations, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

```
device(config-isis-router)# spf-interval level-1 15 10000 15000
```

The following example specifies that the maximum interval in seconds between SPF recalculations is 15 seconds for Level 1 packets. The initial SPF calculation delay is 10000 milliseconds and the hold time between the first and second SPF calculation is 15000 millseconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# spf-interval level-1 15 10000 15000
```

Configuring the IS-IS flooding mechanism

You can configure IS-IS to flood Link State PDUs to other devices in the network before running SPF, thus improving database synchronization by allowing LSP changes to be propagated to neighbors before running SPF. The following task configures the IS-IS fast-flood feature.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **fast-flood** command and specify a value to implement IS-IS fast-flood.

```
device(config-isis-router)# fast-flood 10
```

The following example configures IS-IS to flood 10 LSPs before running SPF.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# fast-flood 10
```

Configuring IS-IS PSPF exponential back-off

The following task changes the partial shortest path first (PSPF) interval, changing the maximum interval between SPF recalculations, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **partial-spf-interval** command with the **level-1** parameter, specifying values for the *max-wait* , *initial-wait* , and *second-wait* parameters, to set the maximum interval between SPF recalculationss, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

```
device(config-isis-router)# partial-spf-interval 15 10000 15000
```

The following example specifies that the maximum interval in seconds between SPF recalculations is 15 seconds. The initial SPF calculation delay is 10000 milliseconds and the hold time between the first and second SPF calculation is 15000 milliseconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# partial-spf-interval 15 10000 15000
```

Hello padding

By default, a device adds extra data to the end of a hello packet to make the packet the same size as the maximum length of PDU the device supports.

The padding applies to the following types of hello packets:

- ES hello (ESH PDU)
- IS hello (ISH PDU)
- IS to IS hello (IIH PDU)

The padding consists of arbitrarily valued octets. A padded hello PDU indicates the largest PDU that the device can receive. Other ISs that receive a padded hello PDU from the device can, therefore, ensure that the IS-IS PDUs they send the device are lower than the maximum value available. Similarly, if the device receives a padded hello PDU from a neighbor IS, the device knows the maximum size PDU that can be sent to the neighbor.

When padding is enabled, the maximum length of a Hello PDU sent by the device is 1514 bytes.

If you need to disable hello padding, you can do so globally or on individual interfaces. Generally, you do not need to disable padding unless a link is experiencing slow performance. If you enable or disable padding on an interface, the interface setting overrides the global setting.

Disabling hello padding globally

By default, a device adds extra data to the end of a hello packet to make the packet the same size as the maximum length of PDU the device supports. The following task disables hello padding globally.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **hello padding** command with the **disable** parameter to disable hello padding globally.

```
device(config-isis-router)# hello padding disable
```

The following example disables hello padding globally.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# hello padding disable
```

Partial SPF optimizations

IS-IS employs certain partial SPF optimizations to make partial changes to the routing table in network change situations where the topology of the network has not changed but where there may be changes in the IP networks advertised.

These optimizations are termed partial SPF optimizations. IS-IS can be configured to perform a full SPF calculation when any network (non-topology) change occurs so that IS-IS always runs full SPF for all such network changes.

> NOTE
> If you disable the partial SPF optimizations, IS-IS automatically disables the incremental SPF optimizations and always runs full SPF. However, the reverse is not true: disabling incremental SPF optimizations does not disable partial optimizations.

Disabling partial SPF optimizations

You can configure IS-IS to perform a full SPF calculation when any network (non-topology) change occurs so that IS-IS always runs full SPF for all such network changes. This task disables partial SPF optimizationss.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **disable-partial-spf-opt** command to disable partial SPF optimizations.

```
device(config-isis-router)# disable-partial-spf-opt
```

The following example disables partial SPF optimizations.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# disable-partial-spf-opt
```

Incremental SPF optimizations

In the event of certain topology changes, for instance non-local adjacency flaps, IS-IS employs incremental SPF optimizations to efficiently update the routing table. An incremental SPF is faster and takes fewer CPU cycles than a full SPF.

IS-IS can be configured to perform a full SPF calculation when any network topology change occurs so that IS-IS always runs full SPF for all such network changes.

NOTE
If you disable the partial SPF optimizations, IS-IS automatically disables the incremental SPF optimizations and always runs full SPF. However, the reverse is not true: disabling incremental SPF optimizations does not disable partial optimizations.

Disabling incremental shortcut SPF optimizations

You can configure IS-IS to perform a full SPF calculation when any network topology change occurs so that IS-IS always runs full SPF for all such network changes. This task disables incremental SPF optimizations so that IS-IS always runs full SPF for all such network changes.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **disable-partial-spf-opt** command to disable partial SPF optimizations.

```
device(config-isis-router)# disable-incremental-spf-opt
```

The following example disables incremental SPF optimizations.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# disable-incremental-spf-opt
```

IS-IS incremental shortcut LSP SPF optimization

IS-IS can be configured to use an incremental shortcut LSP SPF optimization algorithm. Incremental shortcut LSP SPF optimization is more efficient when updating the routes in cases where the shortcut LSP state change does not influence the topology. Incremental Shortcut LSP SPF Optimizations are enabled by default.

NOTE
If you disable the partial SPF optimizations, IS-IS automatically disables the incremental SPF optimizations and always runs full SPF. However, the reverse is not true: disabling incremental SPF optimizations does not disable partial optimizations.

NOTE
Incremental Shortcut SPF optimizations are not applicable to LSP shortcuts with metrics configured on them.

NOTE
Incremental Shortcut SPF optimizations are not applicable to LSP shortcuts with negative relative metrics configured.

NOTE
Incremental Shortcut SPF optimizations are not applicable to announced LSP shortcuts.

Disabling incremental SPF optimizations

IS-IS is configured to use an incremental shortcut LSP SPF optimization algorithm by default. This task disables incremental shortcut LSP SPF optimization.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **disable-inc-stct-spf-opt** command to disable incremental shortcut LSP SPF optimization.

```
device(config-isis-router)# disable-inc-stct-spf-opt
```

The following example disables incremental shortcut LSP SPF optimization.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# disable-inc-stct-spf-opt
```

Maximum number of load sharing paths

You can change the number of paths IPv4 IS-IS can calculate and install in the IPv4 forwarding table. If you change the number of paths to one, the device does not load share multiple route paths learned from IPv4 IS-IS.

By default, IPv4 IS-IS can calculate and install eight equal-cost paths into the IPv4 forwarding table.

Changing the maximum number of load sharing paths

You can specify the number of paths IS-IS can calculate and install in the IP forwarding table. The following task specifies that the number of paths IS-IS can calculate and install in the IP forwarding table is 7.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

```
device(config-isis-router)# address-family ipv4 unicast
```

4. Enter the **maximum-paths** command and specify a value to specify the number of paths IS-IS can calculate and install in the IP forwarding table.

```
device(config-router-isis-ipv4u)# maximum-paths 7
```

The following example specifies that the number of paths IS-IS can calculate and install in the IP forwarding table is 7.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# maximum-paths 7
```

Default route advertisement

By default, a device does not generate or advertise a default route to its neighboring ISs. A default route is not advertised even if the device's IPv4 route table contains a default route. You can enable the device to advertise a default route to all neighboring ISs.

NOTE
This feature requires the presence of a default route in the IPv4 route table.

By default, this feature originates the default route at Level 2 only. However, you can apply a route map to originate the default route to Level 1 only or at both Level 1 and Level 2. The route map must be configured before you can use the route map to configure the device to advertise the default route.

Enabling advertisement of a default route

You can enable the device to advertise a default route to all neighboring ISs. The following task enables the advertisement of a default route.

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv4 unicast
   ```

4. Enter the **default-information-originate** command to enable the advertisement of a default route.

   ```
   device(config-router-isis-ipv4u)# default-information-originate
   ```

The following example enables the advertisement of a default route.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# default-information-originate
```

Using a route map to advertise a default route

You can use a route map to configure the device to advertise a default route to all neighboring ISs. The following task enables the advertisement of a default route if the route map “myroutemap” is satisfied.

The route-map “myroutemap” must already be configured.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

```
device(config-isis-router)# address-family ipv4 unicast
```

4. Enter the **default-information-originate** command with the **route-map** parameter and specify a route-map parameter to use the route map to enable the advertisement of a default route.

```
device(config-router-isis-ipv4u)# default-information-originate route-map myroutemap
```

The following example enables the advertisement of a default route if the route map “myroutemap” is satisfied.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# default-information-originate route-map myroutemap
```

IS-IS administrative distance

When the device has paths from multiple routing protocols to the same destination, it compares the administrative distances of the paths and selects the path with the lowest administrative distance to place in the IPv4 route table.

For example, if the device has a path from iBGP, from OSPF, and IPv4 IS-IS to the same destination, and all the paths are using their protocols’ default administrative distances, the device selects the OSPF path, because that path has a lower administrative distance than the iBGP and IPv4 IS-IS paths.

The default IPv4 administrative distances on the device are:

- Directly connected - 0 (this value is not configurable)
- Static - 1 (applies to all static routes, including default routes)
- eBGP - 20
- OSPF - 110
- IPv4 IS-IS - 115
- iBGP - 200
- Local BGP - 200
- Unknown - 255 (the device will not use this route)

Lower administrative distances are preferred over higher distances. For example, if the device receives routes for the same network from IPv4 IS-IS and from iBGP, it will prefer the IPv4 IS-IS route by default.

Changing the administrative distance for IPv4 IS-IS

You can change the administrative distance for IPv4 IS-IS routes. The following task changes the administrative distance for the IPv4 IS-IS unicast address family.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

```
device(config-isis-router)# address-family ipv4 unicast
```

4. Enter the **distance** command and specify a value to change the administrative distance for the IPv4 IS-IS unicast address family.

```
device(config-router-isis-ipv4u)# distance 60
```

The following example sets an administrative distance of 60 for the IPv4 unicast address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# distance 60
```

Configuring summary addresses

You can configure summary addresses to aggregate IS-IS route information. Summary addresses can enhance performance by reducing the size of the Link State database, reducing the amount of data the device needs to send to its neighbors, and reducing the CPU cycles used for IS-IS. The following task configures a summary address of 10.1.0.0 with a mask of 255.255.0.0 for Level 1 and Level 2 routes.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

```
device(config-isis-router)# address-family ipv4 unicast
```

4. Enter the **summary-address** command with the **level-1** and **level-2** parameters, specifying an IPv4 address and network mask to configure a summary address and network mask for Level 1 and Level 2 routes.

```
device(config-router-isis-ipv4u)# summary-address 10.1.0.0 255.255.0.0 level-1 level-2
```

The following example configures a summary address of 10.1.0.0 with a mask of 255.255.0.0 for Level 1 and Level 2 routes.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# summary-address 10.1.0.0 255.255.0.0 level-1 level-2
```

IPv4 IS-IS route redistribution

Routes can be redistributed into IPv4 IS-IS by performing the following configuration tasks:

- Changing the default redistribution metric (optional).
- Configuring the redistribution of a particular route type into IPv4 IS-IS (mandatory).

The device can redistribute routes from the following route sources into IPv4 IS-IS:

- BGP4+
- OSPF
- Static IPv4 routes
- IPv4 routes learned from directly connected networks

The device can also can redistribute Level 1 IPv4 IS-IS routes into Level 2 IPv4 IS-IS routes, and Level 2 IPv4 IS-IS routes into Level 1 IPv4 IS-IS routes.

Route redistribution from other sources into IPv4 IS-IS is disabled by default. When you enable redistribution, the device redistributes routes only into Level 2 by default. You can specify Level 1 only, Level 2 only, or Level 1 and Level 2 when you enable redistribution.

The device automatically redistributes Level 1 routes into Level 2 routes. Thus, you do not need to enable this type of redistribution. You also can enable redistribution of Level 2 routes into Level 1 routes.

The device attempts to use the redistributed route's metric as the route's IPv4 IS-IS metric. For example, if an OSPF route has an OSPF cost of 20, the device uses 20 as the route's IPv4 IS-IS metric. The device uses the redistributed route's metric as the IPv4 IS-IS metric unless the route does not a have a valid metric. In this case, the device assigns the default metric value to the route.

An IS-IS LSP can hold around 32,000 IPv4 routes or 11,000 IPv6 routes. This number can vary depending on the prefix length of the routes.

Redistributing routes into IPv4 IS-IS

Routes can be redistributed for IPv4-IS-IS, and the routes to be redistributed can be specified.

The redistribution of static routes into IPv4 IS-IS is configured on device1. The redistribution of connected routes into IPv4 IS-IS 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.

```
device1# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device1(config)# router isis
```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

```
device1(config-isis-router)# address-family ipv4 unicast
```

4. Enter the **redistribute** command with the **static** parameter to redistribute static routes.

```
device1(config-router-isis-ipv4u)# redistribute static
```

5. On device2, enter the **configure terminal** command to access global configuration mode.

```
device2# configure terminal
```

6. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device2(config)# router isis
```

7. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

```
device2(config-isis-router)# address-family ipv4 unicast
```

8. Enter the **redistribute** command with the **connected** and **route-map** parameters to redistribute connected routes and specify a route map.

```
device2(config-router-isis-ipv4u)# redistribute connected route-map rmap1
```

The following example redistributes static routes into IPv4 IS-IS on a device.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# redistribute static
```

The following example redistributes connected routes into IPv4 IS-IS on a device and specifies a route map.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# redistribute connected route-map rmap1
```

Default redistribution metric

When IPv4 IS-IS redistributes a route from another route source (such as OSPF, BGP4+, or a static IPv4 route) into IPv4 IS-IS, the route's metric value is used as its metric when the metric is not modified by a route map or metric parameter and the default redistribution metric is set to its default value of 0.

The default metric value can be changed.

NOTE
The implementation of IS-IS does not support the optional metric types Delay, Expense, or Error.

Changing the default redistribution metric

You can change the default metric for the IS-IS routing protocol. The following task changes the default metric value for the IPv4 unicast address family.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

```
device(config-isis-router)# address-family ipv4 unicast
```

4. Enter the **default-metric** command and specify a value to change the default metric value for the IPv4 unicast address family.

```
device(config-router-isis-ipv4u)# default-metric 30
```

The following example sets the default value to 30 for the IPv4 unicast address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# default-metric 30
```

Configuring the default link metric value globally

You can configure the metric value globally on all active IS-IS interfaces for the configured address family. The following task sets the IS-IS default link metric value for Level 1 for the IPv4 address family.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

```
device(config-isis-router)# address-family ipv4 unicast
```

4. Enter the **default-link-metric** command with the **level-1** parameter, and specify a value, to set the IS-IS default link metric value for Level 1 for the IPv4 address family.

```
device(config-router-isis-ipv4u)# default-link-metric level-1 30
```

The following example sets the IS-IS default link metric value for Level 1 for the IPv4 address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# default-link-metric level-1 30
```

IS-IS metric styles

There are two types of metric styles in IS-IS:

- narrow metric
- wide metric

The range of the metric value is different for both of these styles. If there is a change in the metric-style configuration, the default-link-metric changes with it so that the new value of the default-link-metric is equal to the minimum of both the configured value and the maximum value supported for the new metric-style.

If the metric style changes from narrow metric to wide metric, no change in the value of default-link-metric occurs.

If the metric style changes from wide metric to narrow metric, and if the value of default-link-metric is greater than 63, the default-link-metric takes the value 63, as it is the maximum supported in the narrow metric.

Changing the metric style

You can increase the range of metric values supported by the device by changing the metric style to wide. The following task enables the wide metric type for Level 2 packets for the IS-IS IPv4 unicast address-family.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

```
device(config-isis-router)# address-family ipv4 unicast
```

4. Enter the **metric-style wide** command to enable the wide metric type for Level 2 packets for the IS-IS IPv4 unicast address-family.

```
device(config-router-isis-ipv4u)# metric-style wide level-2
```

The following example enables the wide metric type for Level 2 packets for the IS-IS IPv4 unicast address-family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# metric-style wide level-2
```

IS-IS non-stop routing

IS-IS can continue operation without interruption during a processor switchover or hitless-reload event when the Non-Stop Routing (NSR) feature is enabled.

NOTE
IS-IS NSR is supported only on the SLX 9850 with redundant MMs.

In general, a device restart causes its peer to remove the routes originated from the device and reinstalls them. The IS-IS NSR feature enables the device to maintain neighbors and LSA database with its peer in the event of a restart. In IS-IS NSR, the processor switchovers and the hitless-reloads are treated the same as they are during startup and the overload bit is set in the same way as it is after a reboot.

NOTE
IS-IS NSR is independent of Graceful Restart (GR) and GR help role mechanisms.

NOTE
Process restart takes precedence over NSR.

Enabling non-stop routing

IS-IS non-stop routing (NSR) can be re-enabled if it has been disabled. The following task re-enables NSR for IS-IS.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **nonstop-routing** command to re-enable NSR.

```
device(config-isis-router)# nonstop-routing
```

The following example re-enables NSR.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# nonstop-routing
```

Limitations of IS-IS non-stop routing

The following limitations apply for IS-IS non-stop routing (NSR)

- The IS-IS over GRE tunnel feature does not support IS-IS NSR. The GRE tunnel interface types are not supported.
- The IS-IS shortcuts are not supported because they depend on the MPLS tunnel.
- If the IS-IS hellos are forwarded at Layer 2 and the device executes a hitless-reload, hellos will not be forwarded for a brief time. The IS-IS adjacencies are lost for 12 seconds and there will be data traffic loss.
- The configuration events that occur close to switchover or hitless-reload may get lost due to CLI synchronization issues.
- The neighbor or interface state changes close to switchover or hitless-reload cannot be handled.
- The IS-IS neighbor hold timer is restarted upon IS-IS NSR switchover or hitless-reload.
- It is recommended to use the default IS-IS hello timer value. IS-IS neighbor sessions may flap during a linecard software upgrade if a shorter timer value is used.
- The traffic counters are not synchronized because the neighbor and LSP database counters are recalculated on the standby module during synchronization.
- With IS-IS NSR enabled, after switchover or hitless-reload to standby MP, IS-IS routes, LSP database and neighbor adjacencies are maintained so that there will be no loss of existing traffic to the IS-IS destinations.
- The IS-IS NSR hitless failover event may not be completely invisible to the network because, after switchover, additional flooding of CSNP packets will occur in the directly connected neighbors.

DIS hello interval

The DIS hello interval value is derived from the hello interval configured under the interface.

The default IS-IS hello interval is 10 sec. The default DIS hello interval is 10/3 = sec. The default value of the DIS hello interval is not changed. However, if you configure a hello interval of 20 for an interface, then the DIS hello interval for the interface becomes 20/3 =6 sec.

If IS-IS NSR is enabled, the DIS hello is changed to hello timer /2.

The DIS hello multipler is the same as the hello multipler configured under the interface.

Enabling IS-IS for an Interface

IS-IS can be enabled for a specified interface for a device. Several commands can then be accessed that allow the configuration of IS-IS on the specified interface. This task enables IS-IS routing for an interface Ethernet.

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **ip router isis** command to enable IS-IS for the interface.

```
device(conf-if-eth-1/2)# ip router isis
```

The following example enables IS-IS routing for an interface Ethernet.

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

Configuring authentication on an IS-IS interface

The following task configures IS-IS authentication mode, an IS-IS authentication key, and temporarily disables IS-IS authentication checking for Level 2 packets on an IS-IS Ethernet interface.

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **isis auth-mode** command with the **md5** and **level-2** parameters to configure MD5 authentication for Level 2 packets for the IS-IS interface.

```
device(conf-if-eth-1/2)# isis auth-mode MD5 level-2
```

4. Enter the **isis auth-key** command with the **level-2** parameter, specifying an authentication password, to configure an authentication key for Level 2 packets for the IS-IS interface.

```
device(conf-if-eth-1/2)# isis auth-key level-2 mykey
```

5. Enter the **isis auth-check disable** command with the **level-1** parameter to temporarily disable authentication checking for Level 1 and Level 2 packets for the IS-IS interface.

```
device(conf-if-eth-1/2)# isis auth-check level-1 disable
```

The following example configures IS-IS authentication mode, an IS-IS authentication key and temporarily disables IS-IS authentication checking for a specified IS-IS interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis auth-mode MD5 level-2
device(conf-if-eth-1/2)# isis auth-key level-2 mykey
device(conf-if-eth-1/2)# isis auth-check level-1 disable
```

Disabling hello padding for an IS-IS interface

By default, a device adds extra data to the end of a hello packet to make the packet the same size as the maximum length of PDU the device supports. The following task disables hello padding for an IS-IS Ethernet interface.

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

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **isis hello padding** command with the **disable** parameter to disable hello padding for the IS-IS Ethernet interface.

   ```
   device(conf-if-eth-1/2)# isis hello padding disable
   ```

The following example disables hello padding for a specified IS-IS Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis hello padding disable
```

Changing the IS-IS level on an IS-IS interface

By default, a device can operate as both a Level 1 and Level 2 router. The following task changes the level supported from Level 1 and Level 2 to Level 1 for an IS-IS interface.

If you change the IS-IS type on an individual interface, the type you specify must also be specified globally. For example, if you globally set the type to Level 2 only, you cannot set the type on an individual interface to Level 1. The setting is accepted but does not take effect.

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

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **isis circut-type** command with the **level-1** parameter to configure a Level 1 adjacency for the interface.

   ```
   device(config-if-e1000-1/2)# isis circut-type level-1
   device(conf-if-eth-1/2)# isis circut-type level-1
   ```

The following example configures a Level 1 adjacency on an IS-IS Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis circut-type level-1
```

Changing the hello multiplier for an IS-IS interface

The hello multiplier is the number by which an IS-IS interface multiplies the hello interval to obtain the hold time for Level 1 and Level 2 IS-to-IS hello PDUs. The following task changes the hello multiplier for Level 2 packets for an Ethernet interface.

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

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **isis hello-mulitplier** command with the **level-2** parameter, and specify a value, to change the hello multiplier for Level 2 packets for the interface.

   ```
   device(conf-if-eth-1/2)# isis hello-mulitplier level-2 20
   ```

The following example changes the hello multiplier to 20 for Level 2 packets for an Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis hello-mulitplier level-2 20
```

Changing the hello interval for an IS-IS interface

The hello interval specifies how often an IS-IS interface sends hello messages to its IS-IS neighbors. The following task changes the hello interval for Level 1 packets to 20 for an IS-IS interface.

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

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **isis hello-interval** command with the **level-1** parameter, and specify a value, to change the hello interval for Level 1 packets for the interface.

   ```
   device(conf-if-eth-1/2)# isis hello-interval level-1 20
   ```

The following example changes the hello interval for Level 1 packets to 20 on a Loopback interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis hello-interval level-1 20
```

IS-IS point-to-point over Ethernet

IS-IS uses its neighbor's MAC address to form an adjacency and stores the neighbors MAC address to recognize the adjacency in the future. This causes no problems with directly adjacent devices but problems can occur when adjacency is required between devices that are more than one hop away. IS-IS Point-to-Point over Ethernet accommodates an IS-IS network with this type of configuration.

When IS-IS Point-to-Point over Ethernet is used, devices that are several hops away or available through an IP GRE tunnel can form an IS-IS adjacency. IS-IS Point-to-Point over Ethernet can also be used when only two IS's are part of the broadcast network. IS-IS Point-to-Point over Ethernet is configured at the interface level of the devices that are forming an adjacency.The figure below shows two devices several hops away from each other that are configured for IS-IS adjacency.

FIGURE 37 IS-IS Point-to-Point configuration

Enabling IS-IS point-to-point over Ethernet

IS-IS Point-to-Point over Ethernet can be configured so that devices that are several hops away can form an IS-IS adjacency. The following task configures two devices, Device A and Device B, for IS-IS point-to-point over Ethernet.

1. On Device A, enter the **configure terminal** command to access global configuration mode.

```
deviceA# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **ip router isis** command to enable IS-IS for the interface.

```
deviceA(conf-if-eth-1/2)# ip router isis
```

4. Enter the **ip address** command and specify an IP address and mask to configure a primary IP address for the interface.

```
deviceA(conf-if-eth-1/2)# ip address 10.10.1.1/31
```

5. Enter the **isis point-to-point** command to configure IS-IS point-to-point for the Ethernet interface.

```
deviceA(conf-if-eth-1/2)# isis point-to-point
```

6. On Device B, enter the **configure terminal** command to access global configuration mode.

```
deviceB# configure terminal
```

7. Enter the **interface ethernet** command and specify an interface.

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

8. Enter the **ip router isis** command to enable IS-IS for the interface.

```
deviceB(conf-if-eth-2/1)# ip router isis
```

9. Enter the **ip address** command and specify an IP address and mask to configure a primary IP address for the interface.

```
deviceB(conf-if-eth-2/1)# ip address 10.10.1.2/31
```

10. Enter the **isis point-to-point** command to configure IS-IS point-to-point for the Ethernet interface.

```
deviceB(conf-if-eth-2/1)# isis point-to-point
```

The following example configures two devices, Device A and Device B, for IS-IS point-to-point over Ethernet.

```
deviceA# configure terminal
deviceA(config)# interface ethernet 1/2
deviceA(conf-if-eth-1/2)# ip router isis
deviceA(conf-if-eth-1/2)# ip address 10.10.1.1/31
deviceA(conf-if-eth-1/2)# isis point-to-point


deviceB# configure terminal
deviceB(config)# interface ethernet 2/1
deviceB(conf-if-eth-2/1)# ip router isis
deviceB(conf-if-eth-2/1)# ip address 10.10.1.2/31
deviceB(conf-if-eth-2/1)# isis point-to-point
```

Displaying IS-IS statistics

Various **show isis** commands verify information about IPv4 and IPv6 IS-IS configurations.

Use one or more of the following commands to verify IS-IS information. The commands do not have to be entered in this order.

1. Enter the **show isis** command.

```
device# show isis

IS-IS Routing Protocol Operation State: Enabled
IS-Type: Level-1-2
System Id: 768e.f805.5812
Manual area address(es): 11
Level-1-2 Database State: On
Administrative Distance 115
Maximum Paths 8
Default redistribution metric 0
Default link metric for level-1 0 (conf)/ 10 (adv)
Default link metric for level-2 0 (conf)/ 10 (adv)
Protocol Routes Redistributed into IS-IS: None
Number of Routes Redistributed into IS-IS: 0
Level-1 Auth-mode: None
Level-2 Auth-mode: None
Metric Style Supported for Level-1: Narrow
Metric Style Supported for Level-2: Narrow
Graceful-Restart Helper Support: Enabled
ISIS Partial SPF Optimizations: Enabled
Timers:
L1 SPF: Max-wait 5s Init-wait 5000ms Second-wait 5000ms
L2 SPF: Max-wait 5s Init-wait 5000ms Second-wait 5000ms
L1 SPF is not scheduled
L2 SPF is not scheduled
PSPF: Max-wait 5000ms Init-wait 2000ms Second-wait 5000ms
PSPF is not scheduled
LSP: max-lifetime 1200s refresh-interval 900s gen-interval 10s
retransmit-interval 5s, lsp-interval 33ms
SNP: csnp-interval 10s psnp-interval 2s
Global Hello Padding: Enabled
Global Hello Padding For Point to Point Circuits: Enabled
Ptpt Three Way HandShake Mechanism: Enabled
BGP Ipv4 Converged: False BGP Ipv6 Converged: False
IS-IS Traffic Engineering Support: Disabled
No ISIS Shortcuts Configured
BFD: Disabled, BFD HoldoverInterval: 0
NSR: Disabled
LSP-SYNC: Not Globally Enabled
```

   This example output displays general IS-IS information..

2. Enter the **show isis config** command.

```
device# show isis config

router isis
net 11.768e.f805.5812.00
address-family ipv4 unicast
!
address-family ipv6 unicast
!
```

   This example shows the global IS-IS configuration commands that are in effect on the device..

3. Enter the **show isis counts** command.

```
device# show isis counts

Area Mismatch: 0
Max Area Mismatch: 0
System ID Length Mismatch: 0
LSP Sequence Number Skipped: 0
LSP Max Sequence Number Exceeded: 0
Level-1 Database Overload: 1
Level-2 Database Overload: 0
Our LSP Purged: 2
```

This example shows IS-IS error statistics..

4. Enter the **show isis hostname** command.

```
device# show isis hostname

Total number of entries in IS-IS Hostname Table: 1
System ID Hostname * = local IS
-----------------------------------------------------
* 768e.f805.5812 R1
```

This example shows the router-name-to-system-ID mapping table entries for the device.

5. Enter the **show isis interface** command.

```
device# show isis interface

Total number of IS-IS Interfaces: 11
Interface: Ve 301
Circuit State: UP Circuit Mode: Level 1-2
Circuit Type: BCAST Passive State: FALSE
Circuit Number: 2, MTU: 1500
Level-1 Auth-mode: NONE
Level-2 Auth-mode: NONE
Level-1 Metric: 10, Level-1 Priority: 64
Level-1 Hello Interval: 10, Level-1 Hello Multiplier: 3
Level-1 Designated IS: R1-02 Level-1 DIS Changes: 2
Level-2 Metric: 10, Level-2 Priority: 64
Level-2 Hello Interval: 10, Level-2 Hello Multiplier: 3
Level-2 Designated IS: R1-02 Level-2 DIS Changes: 2
Next IS-IS LAN Level-2 Hello in 11 seconds
Number of active Level-2 adjacencies: 0
Next IS-IS LAN Level-1 Hello in 1 seconds
Number of active Level-1 adjacencies: 0
Circuit State Changes: 1 Circuit Adjacencies State Changes: 0
Rejected Adjacencies: 0
Circuit Authentication L1 failures: 0
Circuit Authentication L2 failures: 0
Bad LSPs: 0
Control Messages Sent: 7577 Control Messages Received: 0
Hello Padding: Enabled
IP Enabled: TRUE
IP Addresses:
11.2.1.1/30
IPv6 Enabled: FALSE
MPLS TE Enabled: FALSE
BFD Enabled: FALSE
LDP-SYNC: Disabled, State:
…
```

This example shows information about IS-IS interfaces for a device.

6. Enter the **show isis interface** command, using the **brief** keyword.

```
device# show isis interface brief

Total number of IS-IS Interfaces: 11
Interface Type State Mode Passive MTU UpAdj DIS StateChg AdjStateChg
Ve 301 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 302 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 303 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 304 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 305 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 306 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 307 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 308 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 309 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 310 BCAST UP L12 FALSE 1500 0 None 1 0
Lo 1 BCAST UP L12 TRUE 0 0 None 1 0
```

This example shows summarized information about IS-IS interfaces for a device.

7. Enter the **show isis neighbor** command.

```
device# show isis neighbor
```

This example shows IS-IS neighbor information.

8. Enter the **show isis spf-log** command.

```
device# show isis spf-log

ISIS Level-1 SPF Log
When Duration Nodes Count Last-Trigger-LSP Trigger
10h34m13s 0ms 1 10 R1.00-00 Interface State Change
10h34m38s 0ms 1 2 R1.00-00 Interface Config Change
10h34m43s 0ms 1 18 R1.00-00 Interface Config Change
10h34m48s 0ms 1 5 R1.00-00 Interface State Change
ISIS Level-2 SPF Log
When Duration Nodes Count Last-Trigger-LSP Trigger
10h34m13s 0ms 1 10 R1.00-00 Interface State Change
10h34m38s 0ms 1 2 R1.00-00 Interface Config Change
10h34m43s 0ms 1 18 R1.00-00 Interface Config Change
10h34m48s 0ms 1 5 R1.00-00 Interface State Change
```

This example shows IS-IS link-state packet (LSP) logging information.

9. Enter the **show isis traffic** command.

```
show isis traffic

                            Message Received     Message Sent
 Level-1 Hellos             0                    44912
 Level-2 Hellos             0                    44927
 PTP Hellos                 0                    0
 Level-1 LSP                0                    0
 Level-2 LSP                0                    0
 Level-1 CSNP               0                    0
 Level-2 CSNP               0                    0
 Level-1 PSNP               0                    0
 Level-2 PSNP               0                    0
```

This example shows information about IS-IS packet counts..

IS-IS (IPv6)

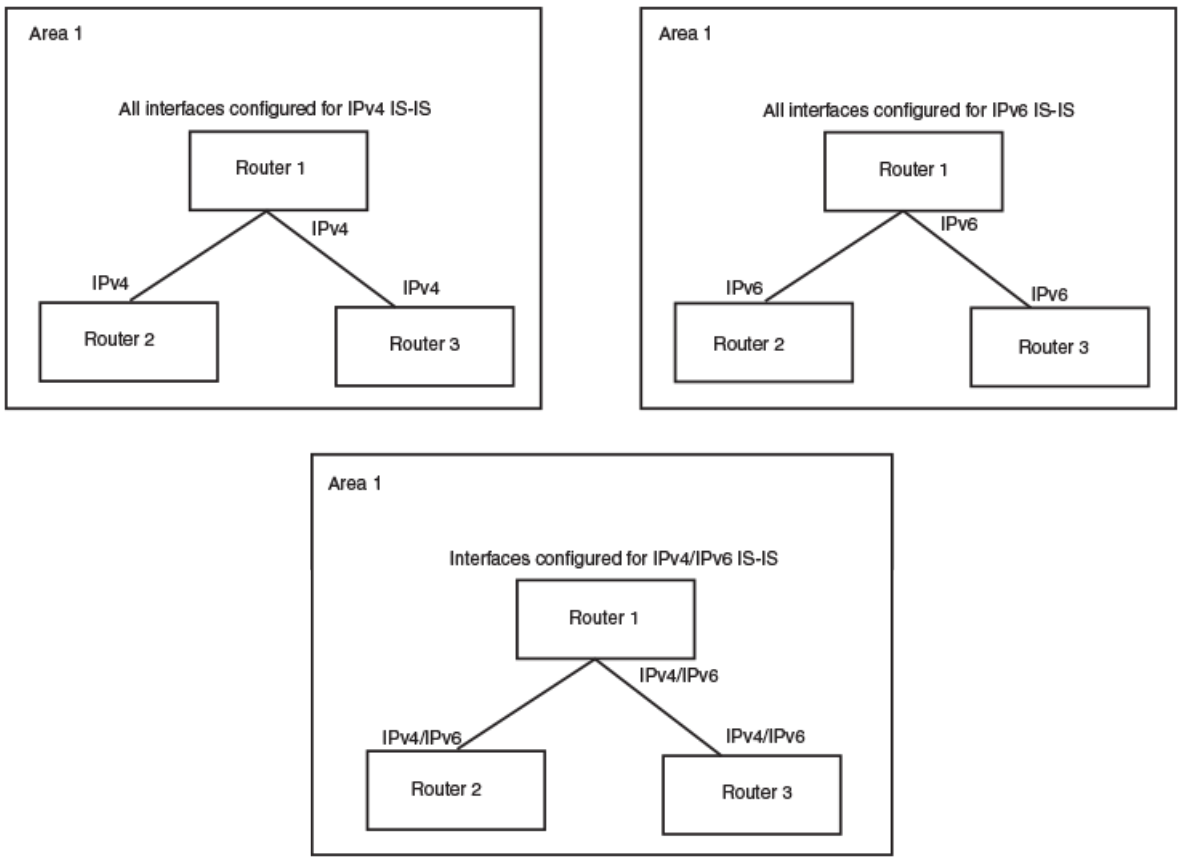
- IPv6 IS-IS single-topology mode 288
- IS-IS CLI levels 289
- Enabling IPv6 IS-IS globally 289
- Configuring the IS-IS IPv6 unicast address family 290
- Configuring IPv6 IS-IS single topology 290
- Setting the overload bit 291
- Configuring authentication 291
- Changing the IS-IS level globally 292
- Logging adjacency changes 292
- Configuring the CSNP interval 293
- Changing the maximum LSP lifetime 293
- Changing the LSP refresh interval 293
- Changing the LSP generation interval 294
- Changing the LSP interval and retransmit interval 294
- Disabling IS-IS name mapping capability 295
- Logging invalid LSP packets received 295
- Changing the SPF timer 296
- Configuring the IS-IS flooding mechanism 296
- Configuring IS-IS PSPF exponential back-off 297
- Disabling hello padding globally 297
- Disabling partial SPF optimizations 298
- Disabling incremental SPF optimizations 298
- Disabling incremental shortcut SPF optimizations 298
- Maximum number of load sharing paths 299
- Default route advertisement 300
- IPv6 IS-IS administrative distance 301
- Configuring summary prefixes 302
- Redistributing routes into IPv6 IS-IS 302
- Default redistribution metric 304
- Configuring the default link metric value globally 305
- IPv6 metric behavior with multi-topology configuration 305
- IS-IS metric styles 305
- IPv6 IS-IS non-stop routing 306
- Limitations of IPv6 IS-IS non-stop routing 306
- IPv6 protocol-support consistency checks 307
- Enabling IS-IS and assigning an IPv6 address to an interface 307
- Configuring authentication on an IS-IS interface 308
- Disabling hello padding for an IS-IS interface 308
- Changing the IS-IS level on an IS-IS interface 309
- Changing the hello multiplier for an IS-IS interface 309
- Changing the hello interval for an IS-IS interface 310
- Changing the metric added to advertised routes for an IS-IS interface 310
- Disabling IPv6 protocol-support consistency checks 311
- IPv6 IS-IS Multi-Topology 311
- Configuration considerations for IPv6 IS-IS MT 312
- Migrating to IPv6 IS-IS MT 312
- Maintaining MT adjacencies 312
- New TLV attributes 313

- Enabling IPv6 IS-IS MT....313
- Configuration example to deploy IPv6 IS-IS MT....313
- Displaying IS-IS statistics....316

IPv6 IS-IS single-topology mode

IPv6 IS-IS supports single-topology mode, which means that you can run IPv6 IS-IS concurrently with other network protocols such as IPv4 IS-IS throughout a topology. However, when implementing a single topology, all routers in an area (Level 1 routing) or domain (Level 2 routing) must be configured with the same set of network protocols on all its interfaces. You can configure IPv4 IS-IS only, IPv6 IS-IS only, or both IPv4 IS-IS and IPv6 IS-IS. For example, to successfully implement both IPv4 and IPv6 IS-IS in an area, you must configure both IPv4 and IPv6 IS-IS on all router interfaces in the area.

FIGURE 38 IPv6 IS-IS in single-topology mode



A single shortest path first (SPF) per level computes the IPv4 and IPv6 routes. The use of a single SPF indicates that both IPv4 and IPv6 IS-IS routing protocols must share a common network topology

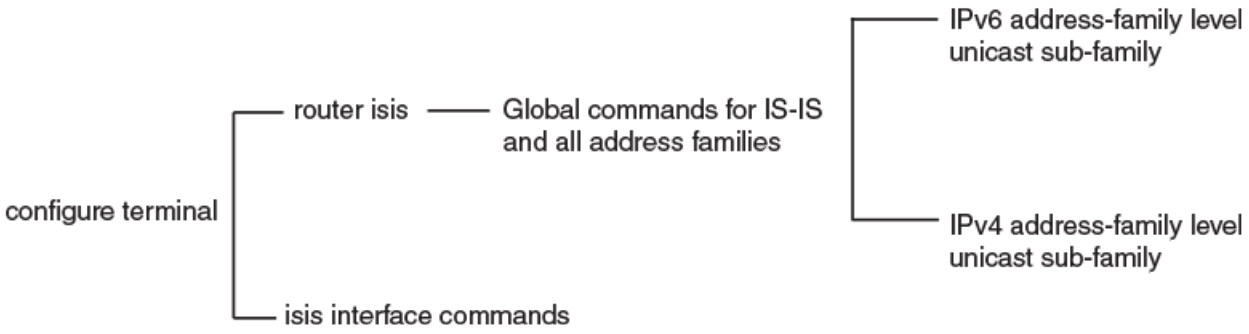
The implementation of IPv4 IS-IS supports type, length, and value (TLV) parameters to advertise reachability to IPv4 networks. The TLVs specify the types of data, the length of the data, and the valid values for the data. IPv6 IS-IS advertises its information using new TLV parameters. The new TLV parameters for IPv6 support an extended default metric value.

In a single topology, if both IPv4 and IPv6 are configured on an interface, metric-style must be set to wide in both address families. Narrow is the default for IPv4. Wide is the default for IPv6.

IS-IS CLI levels

The CLI includes various levels of commands for IS-IS. The figure below shows these levels, including the levels used for IPv6 IS-IS.

FIGURE 39 IPv6 IS-IS CLI levels



The IPv6 IS-IS CLI levels are as follows:

- A global level for the configuration of the IS-IS protocol. At this level, all IS-IS configurations at this level apply to IPv4 and IPv6. Enter this layer using the **router isis** command.
  - Under the global level, you specify an address family. Address families separate the IS-IS configuration IPv6 and IPv4. You enter this level by entering the **address-family** command at the router IS-IS level.
  - Under the address family level, you select a sub-address family, which is the type of routes for the configuration. For IS-IS, you specify **unicast**.
  - An interface level.

The implementation of IPv6 IS-IS includes a new configuration level: ISIS address-family IPv6 unicast configuration mode. You enter IS-IS definitions for IPv6 IS-IS under this level. Address-family allows you to create configurations for IPv6 IS-IS unicast routes that are separate and distinct from configurations for IPv4 IS-IS unicast routes.

NOTE

Each address family configuration level allows you to access commands that apply to that particular address family only. To enable a feature in a particular address family, you must specify any associated commands for that feature in that particular address family. A feature configured in IS-IS IPv4 unicast address family configuration mode does not work for the IPv6 IS-IS unicast address family unless it is explicitly configured in ISIS address-family IPv6 unicast configuration mode. .

Enabling IPv6 IS-IS globally

When IS-IS is enabled on a device, the device enters IS-IS router configuration mode. Several commands can then be accessed that allow the configuration of IS-IS.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
ISIS: Please configure NET!
```

3. Enter the **net** command and specify a NSAP address to configure a NET for IS-IS.

```
device(config-isis-router)# net 49.2211.0000.00bb.cccc.00
```

The following example enables IPv6 IS-IS on a device.

```
device# configure terminal
device(config)# router isis
ISIS: Please configure NET!
device(config-isis-router)# net 49.2211.0000.00bb.cccc.00
```

Configuring the IS-IS IPv6 unicast address family

The IS-IS IPv6 unicast address family allows you to configure IPv6 IS-IS unicast settings that are separate and distinct from IPv4 IS-IS unicast settings. The following task enables ISIS IPv6 address family unicast configuration mode where a variety of IS-IS features can be configured.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

```
device(config-isis-router)# address-family ipv6 unicast
```

The following example enables ISIS address-family IPv6 unicast configuration mode.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)#
```

Configuring IPv6 IS-IS single topology

If your IS-IS single topology supports both IPv6 and IPv4, you can configure both IPv6 and IPv4 on an IS-IS interface for Level 1, Level 2, or both Level 1 and Level 2. However, if you configure both IPv6 and IPv4 on an IS-IS interface, they must be configured to run on the same level. For example, you can configure IPv6 to run on Level 1 on an interface and IPv4 to also run on Level 1 on the same interface. However, you cannot configure IPv6 to run on Level 1 on an interface and IPv4 to run to Level 2 on the same interface.

To configure an IPv6 IS-IS single topology, you must perform the tasks listed below.

1. Globally enable IS-IS and configure at least one Network Entity Title (NET). The NET is the device's network interface with IS-IS. You can configure up to three NETs on a device.
2. Configure the desired device interfaces with an IPv6 address and enable IPv6 IS-IS on the device interfaces.
3. Configure IS-IS parameters.

Setting the overload bit

The overload bit can be configured so that when an IS is overloaded, other ISs will not use the overloaded IS to forward traffic. The following task specifies that the overload bit is set upon system startup and remains set until BGP has converged and specifies that the device that 240 seconds is the maximum time that IS-IS will wait for BGP convergence to complete.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **set-overload-bit** command with the **on-startup** and **wait-for-bgp** parameters, specifying a value, to configure the device to set the overload bit upon system startup. The overload bit remains set until BGP has converged. 86,400 seconds is the maximum time that IS-IS waits for BGP convergence to complete.

```
device(config-isis-router)# set-overload-bit on-startup wait-for-bgp 86400
```

The following example configures the overload bit on system startup until BGP has converged. The maximum time that the device waits for BGP convergence to complete is 86400 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# set-overload-bit on-startup wait-for-bgp 86400
```

Configuring authentication

A device can be configured to authenticate packets sent to or received from an end system (ES) or other intermediate system (IS). The following task configures IS-IS authentication mode, an IS-IS authentication key and temporarily disables IS-IS authentication checking for Level 1 packets.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **auth-mode** command with the **md5** and **level-1** parameters to configure MD5 authentication for Level 1 packets.

```
device(config-isis-router)# auth-mode MD5 level-1
```

4. Enter the **auth-check disable** command with the **level-1** parameter to temporarily disable authentication checking globally for Level 1 packets.

```
device(config-isis-router)# auth-check level-1 disable
```

5. Enter the **auth-key** command with the **level-1** parameters, specifying an authentication password to configure an authentication key globally for Level 1 packets.

```
device(config-isis-router)# auth-key level-1 mysecurekey
```

The following example configures IS-IS authentication mode, an IS-IS authentication key and temporarily disables IS-IS authentication checking for Level 1 packets.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# auth-mode MD5 level-1
device(config-isis-router)# auth-check level-1 disable
device(config-isis-router)# auth-key level-1 mysecurekey
```

Changing the IS-IS level globally

By default, a device can operate as both a Level 1 and Level 2 router. This task globally changes the level supported from Level 1 and Level 2 to Level 1 only.

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **is-type** command with the **level-1** parameter to globally change the IS-IS level supported from Level-1 and Level-2 to Level-1 only.

   ```
   device(config-isis-router)# is-type level-1
   ```

The following example globally changes the IS-IS level supported from Level-1 and Level-2 to Level-1 only.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# is-type level-1
```

Logging adjacency changes

You can configure a device to log changes in the status of an adjacency with another IS. The following task enables the logging of adjacency changes.

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **log adjacency** command to enable the logging of adjacency changes.

   ```
   device(config-isis-router)# log adjacency
   ```

The following example enables the logging of adjacency changes.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# log adjacency
```

Configuring the CSNP interval

By default the Complete Sequence Numbers PDU (CSNP) interval is 10 seconds. The following task configures CSNP interval of 25 seconds for Level 1 and Level 2 packets.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **csnp-interval** command and specify a value to globally configure the CSNP interval.

```
device(config-isis-router)# csnp-interval 25
```

The following example configures a CSNP interval of 25 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# csnp-interval 25
```

Changing the maximum LSP lifetime

This task changes the maximum number of seconds an unrefreshed LSP remains in a device's LSP database from the default of 1200 seconds (20 minutes) to 2400 seconds (40 minutes).

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **max-lsp-lifetime** command and specify a value to globally configure the maximum LSP lifetime.

```
device(config-isis-router)# max-lsp-lifetime 2400
```

The following example changes the maximum LSP lifetime to 2400 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# max-lsp-lifetime 2400
```

Changing the LSP refresh interval

This task changes the maximum number of seconds a device waits between sending updated LSPs to its IS-IS neighbors from the default of 900 seconds to 1800 seconds.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **lsp-refresh-interval** command and specify a value to change the LSP refresh interval.

```
device(config-isis-router)# lsp-refresh-interval 1800
```

The following example changes the LSP refresh interval to 1800 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# lsp-refresh-interval 1800
```

Changing the LSP generation interval

This task changes the minimum number of seconds the device waits between sending updated LSPs to its IS-IS neighbors from the default of 10 seconds to 60 seconds (one minute).

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **lsp-gen-interval** command and specify a value to change the LSP generation interval.

```
device(config-isis-router)# lsp-gen-interval 60
```

The following example changes the LSP generation interval to 60 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# lsp-gen-interval 60
```

Changing the LSP interval and retransmit interval

The LSP interval is the rate of transmission, in milliseconds, of the LSPs. The retransmit interval is the time, in seconds, the device waits before it retransmits LSPs. This task changes the LSP interval to 45 milliseconds and changes the retransmission interval to 10 milliseconds.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **lsp-interval 45** command and specify a value to change the LSP interval from the default of 33 milliseconds.

```
device(config-isis-router)# lsp-interval 42
```

4. Enter the **retransmit-interval** command and specify a value to change the retransmission interval from the default of 5 seconds.

```
device(config-isis-router)# retransmit-interval 10
```

The following example changes the LSP interval to 45 milliseconds and changes the retransmission interval to 10 milliseconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# lsp-interval 42
device(config-isis-router)# retransmit-interval 10
```

Disabling IS-IS name mapping capability

The implementation of IS-IS supports RFC 2763, which describes a mechanism for mapping IS-IS system IDs to the hostnames of the devices with those IDs. The following task disables IS-IS name mapping.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **hostname disable** command to disable IS-IS name mapping.

```
device(config-isis-router)# hostname disable
```

The following example globally disables IS-IS name mapping.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# hostname disable
```

Logging invalid LSP packets received

You can configure a device to provide logging of invalid LSP packets. The following task enables the logging of invalid LSP packets.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **log invalid-lsp-packets** command to enable the logging of invalid LSP packets.

```
device(config-isis-router)# log invalid-lsp-packets
```

The following example enables the logging of invalid LSP packets.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# log invalid-lsp-packets
```

Changing the SPF timer

Every IS maintains a Shortest Path First (SPF) tree, which is a representation of the states of each of the IS's links to ESs and other ISs. If the IS is both a Level-1 and Level-2 IS, it maintains separate SPF trees for each level. To ensure that the SPF tree remains current, the IS updates the tree at regular intervals following a change in network topology or the link state database. This task changes the maximum interval in seconds between SPF recalculations, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **spf-interval** command with the **level-1** parameter, specifying values for the *max-wait* , *initial-wait* , and *second-wait* parameters, to set the maximum interval in seconds between SPF recalculations, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

```
device(config-isis-router)# spf-interval level-1 15 10000 15000
```

The following example specifies that the maximum interval in seconds between SPF recalculations is 15 seconds for Level 1 packets. The initial SPF calculation delay is 10000 milliseconds and the hold time between the first and second SPF calculation is 15000 millseconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# spf-interval level-1 15 10000 15000
```

Configuring the IS-IS flooding mechanism

You can configure IS-IS to flood Link State PDUs to other devices in the network before running SPF, thus improving database synchronization by allowing LSP changes to be propagated to neighbors before running SPF. The following task configures the IS-IS fast-flood feature.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **fast-flood** command and specify a value to implement IS-IS fast-flood.

```
device(config-isis-router)# fast-flood 10
```

The following example configures IS-IS to flood 10 LSPs before running SPF.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# fast-flood 10
```

Configuring IS-IS PSPF exponential back-off

The following task changes the partial shortest path first (PSPF) interval, changing the maximum interval between SPF recalculations, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **partial-spf-interval** command with the **level-1** parameter, specifying values for the *max-wait* , *initial-wait* , and *second-wait* parameters, to set the maximum interval between SPF recalculationss, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

```
device(config-isis-router)# partial-spf-interval 15 10000 15000
```

The following example specifies that the maximum interval in seconds between SPF recalculations is 15 seconds. The initial SPF calculation delay is 10000 milliseconds and the hold time between the first and second SPF calculation is 15000 milliseconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# partial-spf-interval 15 10000 15000
```

Disabling hello padding globally

By default, a device adds extra data to the end of a hello packet to make the packet the same size as the maximum length of PDU the device supports. The following task disables hello padding globally.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **hello padding** command with the **disable** parameter to disable hello padding globally.

```
device(config-isis-router)# hello padding disable
```

The following example disables hello padding globally.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# hello padding disable
```

Disabling partial SPF optimizations

You can configure IS-IS to perform a full SPF calculation when any network (non-topology) change occurs so that IS-IS always runs full SPF for all such network changes. This task disables partial SPF optimizationss.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **disable-partial-spf-opt** command to disable partial SPF optimizations.

```
device(config-isis-router)# disable-partial-spf-opt
```

The following example disables partial SPF optimizations.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# disable-partial-spf-opt
```

Disabling incremental SPF optimizations

IS-IS is configured to use an incremental shortcut LSP SPF optimization algorithm by default. This task disables incremental shortcut LSP SPF optimization.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **disable-inc-stct-spf-opt** command to disable incremental shortcut LSP SPF optimization.

```
device(config-isis-router)# disable-inc-stct-spf-opt
```

The following example disables incremental shortcut LSP SPF optimization.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# disable-inc-stct-spf-opt
```

Disabling incremental shortcut SPF optimizations

You can configure IS-IS to perform a full SPF calculation when any network topology change occurs so that IS-IS always runs full SPF for all such network changes. This task disables incremental SPF optimizations so that IS-IS always runs full SPF for all such network changes.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **disable-partial-spf-opt** command to disable partial SPF optimizations.

```
device(config-isis-router)# disable-incremental-spf-opt
```

The following example disables incremental SPF optimizations.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# disable-incremental-spf-opt
```

Maximum number of load sharing paths

You can change the number of paths IPv6 IS-IS can calculate and install in the IPv6 forwarding table. If you change the number of paths to one, the device does not load share multiple route paths learned from IPv4 IS-IS.

By default, IPv6 IS-IS can calculate and install eight equal-cost paths into the IPv4 forwarding table.

Changing the maximum number of load sharing paths

You can specify the number of paths IS-IS can calculate and install in the IP forwarding table. The following task specifies that the number of paths IS-IS can calculate and install in the IP forwarding table is 5 for the IS-IS IPv6 unicast address family.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

```
device(config-isis-router)# address-family ipv6 unicast
```

4. Enter the **maximum-paths** command and specify a value to specify the number of paths IS-IS can calculate and install in the IP forwarding table.

```
device(config-router-isis-ipv6u)# maximum-paths 5
```

The following example specifies that the number of paths IS-IS can calculate and install in the IP forwarding table is 5 for the IS-IS IPv6 unicast address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# maximum-paths 5
```

Default route advertisement

By default, a device does not generate or advertise a default route to its neighboring ISs. A default route is not advertised even if the device's IPv6 route table contains a default route. You can enable the device to advertise a default route to all neighboring ISs.

NOTE
This feature requires the presence of a default route in the IPv6 route table.

You can also use a route map to configure the device to advertise a default route. The route map must be configured before you can use the route map to configure the device to advertise the default route.

Enabling advertisement of a default route

You can enable the device to advertise a default route to all neighboring ISs. The following task enables the advertisement of a default route.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

```
device(config-isis-router)# address-family ipv6 unicast
```

4. Enter the **default-information-originate** command to enable the advertisement of a default route.

```
device(config-router-isis-ipv6u)# default-information-originate
```

The following example enables the advertisement of a default route for the IS-IS IPv6 unicast address-family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# default-information-originate
```

Using a route map to advertise a default route

You can use a route map to configure the device to advertise a default route to all neighboring ISs. The following task enables the advertisement of a default route if the route map "myroutemap" is satisfied for the IS-IS IPv6 address family.

The route-map "myroutemap" must already be configured.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

```
device(config-isis-router)# address-family ipv6 unicast
```

4. Enter the **default-information-originate** command with the **route-map** parameter and specify a route-map parameter to use the route map to enable the advertisement of a default route.

```
device(config-router-isis-ipv6u)# default-information-originate route-map myroutemap
```

The following example enables the advertisement of a default route if the route map "myroutemap" is satisfied for the IPv6 address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# default-information-originate route-map myroutemap
```

IPv6 IS-IS administrative distance

When the device has paths from multiple routing protocols to the same destination, it compares the administrative distances of the paths and selects the path with the lowest administrative distance to place in the IPv6 route table.

For example, if the device has a path from iBGP, from OSPFv3, and IPv6 IS-IS to the same destination, and all the paths are using their protocols' default administrative distances, the device selects the OSPFv3 path, because that path has a lower administrative distance than the iBGP and IPv6 IS-IS paths.

The default IPv6 administrative distances on the device are:

- Directly connected - 0 (this value is not configurable)
- Static - 1 (applies to all static routes, including default routes)
- eBGP - 20
- OSPFv3 - 110
- IPv6 IS-IS - 115
- iBGP - 200
- Local BGP - 200
- Unknown - 255 (the device will not use this route)

Lower administrative distances are preferred over higher distances. For example, if the device receives routes for the same network from IPv6 IS-IS and from iBGP, it will prefer the IPv6 IS-IS route by default.

Changing the administrative distance for IPv6 IS-IS

You can change the administrative distance for IPv6 IS-IS routes. The following task changes the administrative distance for the IS-IS IPv6 unicast address family.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

```
device(config-isis-router)# address-family ipv6 unicast
```

4. Enter the **distance** command and specify a value to change the administrative distance for the IPv6 IS-IS unicast address family.

```
device(config-router-isis-ipv6u)# distance 60
```

The following example sets an administrative distance of 60 for the IPv6 unicast address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# distance 60
```

Configuring summary prefixes

You can configure summary prefixes to aggregate IS-IS IPv6 route information. Summary prefixes can enhance performance by reducing the size of the Link State database, reducing the amount of data the device needs to send to its neighbors, and reducing the CPU cycles used for IPv6 IS-IS. The following task configures a summary prefix 2001:db8::/32 for Level 1 routes.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

```
device(config-isis-router)# address-family ipv6 unicast
```

4. Enter the **summary-prefix** command with the **level-1** parameter, specifying an IPv6 address and network mask, to configure a summary prefix and network mask for Level 1 routes.

```
device(config-router-isis-ipv6u)# summary-prefix 2001:db8::/32 level-1
```

The following example configures asummary-prefix of 2001:db8::/32 for Level 1 routes.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# summary-prefix 2001:db8::/32 level-1
```

Redistributing routes into IPv6 IS-IS

Routes can be redistributed into IPv6 IS-IS by performing the following configuration tasks:

- Changing the default redistribution metric (optional).
- Configuring the redistribution of a particular route type into IPv6 IS-IS (mandatory).

The device can redistribute routes from the following route sources into IPv6 IS-IS:

- BGP4+
- OSPFv3
- Static IPv6 routes
- IPv6 routes learned from directly connected networks

The device can also can redistribute Level 1 IPv6 IS-IS routes into Level 2 IPv6 IS-IS routes, and Level 2 IPv6 IS-IS routes into Level 1 IPv6 IS-IS routes.

Route redistribution from other sources into IPv6 IS-IS is disabled by default. When you enable redistribution, the device redistributes routes only into Level 2 by default. You can specify Level 1 only, Level 2 only, or Level 1 and Level 2 when you enable redistribution.

The device automatically redistributes Level 1 routes into Level 2 routes. Thus, you do not need to enable this type of redistribution. You also can enable redistribution of Level 2 routes into Level 1 routes.

The device attempts to use the redistributed route's metric as the route's IPv6 IS-IS metric. For example, if an OSPFv3 route has an OSPF cost of 20, the device uses 20 as the route's IPv6 IS-IS metric. The device uses the redistributed route's metric as the IPv6 IS-IS metric unless the route does not a have a valid metric. In this case, the device assigns the default metric value to the route.

An IS-IS LSP can hold around 32,000 IPv4 routes or 11,000 IPv6 routes. This number can vary depending on the prefix length of the routes.

Redistributing routes into IPv6 IS-IS

Routes can be redistributed for IPv6-IS-IS, and the routes to be redistributed can be specified.

The redistribution of static routes into IPv6 IS-IS is configured on device1. The redistribution of BGP routes into IPv6 IS-IS is configured on device2, and the BGP routes to be redistributed are specified.

1. On device1, enter the **configure terminal** command to access global configuration mode.

   ```
   device1# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device1(config)# router isis
   ```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

   ```
   device1(config-isis-router)# address-family ipv6 unicast
   ```

4. Enter the **redistribute** command with the **static** parameter to redistribute static routes.

   ```
   device1(config-router-isis-ipv6u)# redistribute static
   ```

5. On device2, enter the **configure terminal** command to access global configuration mode.

   ```
   device2# configure terminal
   ```

6. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device2(config)# router isis
   ```

7. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

   ```
   device2(config-isis-router)# address-family ipv6 unicast
   ```

8. Enter the **redistribute** command with the **bgp** and **route-map** parameters to redistribute BGP routes and specify a route map.

   ```
   device2(config-router-isis-ipv6u)# redistribute bgp route-map myroutemap
   ```

The following example redistributes static routes into IPv6 IS-IS on a device.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# redistribute static
```

The following example redistributes BGP routes into IPv6 IS-IS on a device and specifies a route map.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# redistribute bgp route-map myroutemap
```

Default redistribution metric

When IPv6 IS-IS redistributes a route from another route source (such as OSPv3F, BGP4+, or a static IPv6 route) into IPv6 IS-IS, the route's metric value is used as its metric when the metric is not modified by a route map or metric parameter and the default redistribution metric is set to its default value of 0.

The default metric value can be changed.

NOTE
The implementation of IS-IS does not support the optional metric types Delay, Expense, or Error.

Changing the default redistribution metric

You can change the default metric for the IS-IS routing protocol. The following task changes the default metric value for the IPv6 unicast address family.

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv6 unicast
   ```

4. Enter the **default-metric** command and specify a value to change the default metric value for the IPv6 unicast address family.

   ```
   device(config-router-isis-ipv6u)# default-metric 45
   ```

The following example sets the default value to 45 for the IPv6 unicast address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# default-metric 45
```

Configuring the default link metric value globally

You can configure the metric value globally on all active IS-IS interfaces for the IS-IS IPv6 address family. The following task sets the IS-IS default link metric value for Level 1 for the IPv6 address family.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

```
device(config-isis-router)# address-family ipv6 unicast
```

4. Enter the **default-link-metric** command with the **level-1** parameter, and specify a value, to set the IS-IS default link metric value for Level 1 for the IPv6 address family.

```
device(config-router-isis-ipv6u)# default-link-metric level-1 45
```

The following example sets the IS-IS default link metric value for Level 1 for the IPv6 address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# default-link-metric level-1 45
```

IPv6 metric behavior with multi-topology configuration

The default-link-metric for IPv6 depends upon the multi-topology configuration.

No multi-topology: The IPv6 default-link-metric is the same as that configured for IPv4 address-family.

Multi-topology: The IPv6 default-link-metric is equal to the value configured for IPv6 address-family.

Multi-topology transition: The IPv6 default-link-metric will be equal to the value configured for IPv6 address-family.

IS-IS metric styles

There are two types of metric styles in IS-IS:

- narrow metric
- wide metric

The range of the metric value is different for both of these styles. If there is a change in the metric-style configuration, the default-link-metric changes with it so that the new value of the default-link-metric is equal to the minimum of both the configured value and the maximum value supported for the new metric-style.

If the metric style changes from narrow metric to wide metric, no change in the value of default-link-metric occurs.

If the metric style changes from wide metric to narrow metric, and if the value of default-link-metric is greater than 63, the default-link-metric takes the value 63, as it is the maximum supported in the narrow metric.

IPv6 IS-IS non-stop routing

IPv6 IS-IS can continue operation without interruption during a processor switchover or hitless-reload event when the non-stop routing (NSR) feature is enabled.

NOTE
IS-IS NSR is supported only on the SLX 9850 with redundant MMs.

In general, a device restart causes its peer to remove the routes originated from the router and reinstalls them. The IPv6 IS-IS NSR feature enables the device to maintain neighbors and LSA database with its peer in the event of a restart. In IS-IS NSR, the processor switchovers and the hitless-reloads are treated the same as they are during startup and the overload bit is set in the same way as it is after a reboot. IPv6 IS-IS NSR is compatible with IPv4 IS-IS NSR.

NOTE
IPv6 IS-IS NSR is independent of Graceful Restart (GR) and GR help role mechanisms.

NOTE
Process restart takes precedence over NSR.

Enabling non-stop routing

IS-IS non-stop routing (NSR) can be re-enabled if it has been disabled. The following task re-enables NSR for IS-IS.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **nonstop-routing** command to re-enable NSR.

```
device(config-isis-router)# nonstop-routing
```

The following example re-enables NSR.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# nonstop-routing
```

Limitations of IPv6 IS-IS non-stop routing

- The IS-IS over GRE tunnel feature does not support IS-IS NSR. The GRE tunnel interface types are not supported.
- The IS-IS shortcuts are not supported because they depend on the MPLS tunnel.
- If the IS-IS hellos are forwarded at Layer 2 and the device executes a hitless-reload, hellos will not be forwarded for a brief time. The IS-IS adjacencies are lost for 12 seconds and there will be data traffic loss.
- The configuration events that occur close to switchover or hitless-reload may get lost due to CLI synchronization issues.
- The neighbor or interface state changes close to switchover or hitless-reload cannot be handled.
- The IS-IS neighbor hold timer is restarted upon IS-IS NSR switchover or hitless-reload.

- It is recommended to use the default IS-IS hello timer value. IS-IS neighbor sessions may flap during a linecard software upgrade if a shorter timer value is used.
- The traffic counters are not synchronized because the neighbor and LSP database counters are recalculated on the standby module during synchronization.
- When IS-IS NSR is enabled, after switchover or hitless-reload to standby MP, IS-IS routes, LSP database and neighbor adjacencies are maintained so that there will be no loss of existing traffic to the IS-IS destinations.
- The IS-IS NSR hitless failover event may not be completely invisible to the network because, after switchover, additional flooding of CSNP packets will occur in the directly connected neighbors.

IPv6 protocol-support consistency checks

An IS-IS single topology must be configured to run the same set of network protocols (IPv4 IS-IS only, IPv6 IS-IS only, or both IPv4 IS-IS and IPv6 IS-IS).

By default, IS-IS performs consistency checks on hello packets. If a hello packet does not have the same configured network protocols, IS-IS rejects the packet. For example, a hello packet from a device running IPv4 and IPv6 IS-IS will be rejected by a device running either IPv4 IS-IS only or IPv6 IS-IS only, and the two devices will not become adjacent.

You can configure two devices running different sets of network protocols to form an adjacency.

Enabling IS-IS and assigning an IPv6 address to an interface

IS-IS can be enabled for a specified interface for a device. Several commands can then be accessed that allow the configuration of IS-IS on the specified interface. This task enables IS-IS routing for an interface Ethernet.

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **ipv6 address** command with the **eui-64** parameter, specifying an IPv6 address, to assign the IPv6 address to the interface and configure the IPv6 address with an EUI-64 interface ID in the low-order 64 bits.

```
device(conf-if-eth-1/2)# ipv6 address 2001:db8:12d:1300::/64 eui-64
```

4. Enter the **ipv6 router isis** command to enable IS-IS for the interface.

```
device(conf-if-eth-1/2)# ipv6 router isis
```

The following example enables IS-IS routing for an interface Ethernet and assigns an IPv6 address to the interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# ipv6 address 2001:db8:12d:1300::/64 eui-64
device(conf-if-eth-1/2)# ipv6 router isis
```

Configuring authentication on an IS-IS interface

The following task configures IS-IS authentication mode, an IS-IS authentication key, and temporarily disables IS-IS authentication checking for Level 2 packets on an IS-IS Ethernet interface.

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **isis auth-mode** command with the **md5** and **level-2** parameters to configure MD5 authentication for Level 2 packets for the IS-IS interface.

```
device(conf-if-eth-1/2)# isis auth-mode MD5 level-2
```

4. Enter the **isis auth-key** command with the **level-2** parameter, specifying an authentication password, to configure an authentication key for Level 2 packets for the IS-IS interface.

```
device(conf-if-eth-1/2)# isis auth-key level-2 mykey
```

5. Enter the **isis auth-check disable** command with the **level-1** parameter to temporarily disable authentication checking for Level 1 and Level 2 packets for the IS-IS interface.

```
device(conf-if-eth-1/2)# isis auth-check level-1 disable
```

The following example configures IS-IS authentication mode, an IS-IS authentication key and temporarily disables IS-IS authentication checking for a specified IS-IS interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis auth-mode MD5 level-2
device(conf-if-eth-1/2)# isis auth-key level-2 mykey
device(conf-if-eth-1/2)# isis auth-check level-1 disable
```

Disabling hello padding for an IS-IS interface

By default, a device adds extra data to the end of a hello packet to make the packet the same size as the maximum length of PDU the device supports. The following task disables hello padding for an IS-IS Ethernet interface.

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **isis hello padding** command with the **disable** parameter to disable hello padding for the IS-IS Ethernet interface.

```
device(conf-if-eth-1/2)# isis hello padding disable
```

The following example disables hello padding for a specified IS-IS Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis hello padding disable
```

Changing the IS-IS level on an IS-IS interface

By default, a device can operate as both a Level 1 and Level 2 router. The following task changes the level supported from Level 1 and Level 2 to Level 1 for an IS-IS interface.

If you change the IS-IS type on an individual interface, the type you specify must also be specified globally. For example, if you globally set the type to Level 2 only, you cannot set the type on an individual interface to Level 1. The setting is accepted but does not take effect.

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

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **isis circut-type** command with the **level-1** parameter to configure a Level 1 adjacency for the interface.

   ```
   device(config-if-e1000-1/2)# isis circut-type level-1

   device(conf-if-eth-1/2)# isis circut-type level-1
   ```

The following example configures a Level 1 adjacency on an IS-IS Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis circut-type level-1
```

Changing the hello multiplier for an IS-IS interface

The hello multiplier is the number by which an IS-IS interface multiplies the hello interval to obtain the hold time for Level 1 and Level 2 IS-to-IS hello PDUs. The following task changes the hello multiplier for Level 2 packets for an Ethernet interface.

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

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **isis hello-mulitplier** command with the **level-2** parameter, and specify a value, to change the hello multiplier for Level 2 packets for the interface.

   ```
   device(conf-if-eth-1/2)# isis hello-mulitplier level-2 20
   ```

The following example changes the hello multiplier to 20 for Level 2 packets for an Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis hello-mulitplier level-2 20
```

Changing the hello interval for an IS-IS interface

The hello interval specifies how often an IS-IS interface sends hello messages to its IS-IS neighbors. The following task changes the hello interval for Level 1 packets to 20 for an IS-IS interface.

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **isis hello-interval** command with the **level-1** parameter, and specify a value, to change the hello interval for Level 1 packets for the interface.

```
device(conf-if-eth-1/2)# isis hello-interval level-1 20
```

The following example changes the hello interval for Level 1 packets to 20 on a Loopback interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis hello-interval level-1 20
```

Changing the metric added to advertised routes for an IS-IS interface

When a device originates an IS-IS route or calculates a route, it adds a metric (cost) to the route. Each IS-IS interface has a separate metric value. The following task changes the IS-IS metric for an IS-IS Ethernet interface.

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

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

3. Enter the **isis metric** command with the **level-1** parameter and specify a value to change the metric for the interface, specifying Level 1 packets.

```
device(conf-if-eth-1/2)# isis metric 38 level-1
```

The following example changes the metric for an IS-IS Ethernet interface to 38, specifying Level 1 packets. .

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis metric 38 level-1
```

Disabling IPv6 protocol-support consistency checks

By default, IS-IS performs consistency checks on hello packets. If a hello packet does not have the same configured network protocols, IS-IS rejects the packet. The following task disables the IS-IS IPv6 protocol-support consistency checks so that two devices running different sets of network protocols can form an adjacency.

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

```
device(config-isis-router)# address-family ipv6 unicast
```

4. Enter the **disable-adjacency-check** command to disable IS-IS IPv6 protocol-support consistency checks.

```
device(config-router-isis-ipv6u)# disable-adjacency-check
```

The following example disables IS-IS IPv6 protocol-support consistency checks..

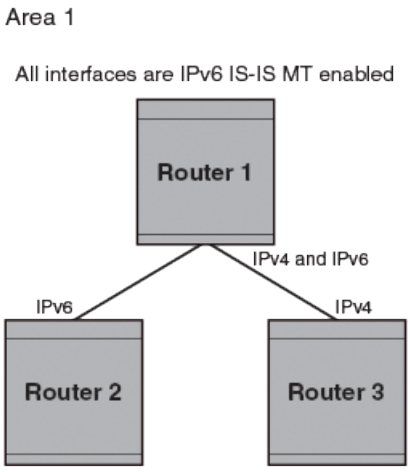
```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# disable-adjacency-check
```

IPv6 IS-IS Multi-Topology

IPv6 IS-IS supports Multi-Topology (MT) mode, which allows you to configure both IPv4 and IPv6 topologies on the router interfaces in an area or a domain. However, when implementing an MT, all routers in an area (Level 1 routing) or a domain (Level 2 routing) can be configured with a set of independent topologies on all their interfaces, even on loopback interfaces. All routers in an area or a domain use the same type of IPv6 support, either single-topology or MT. In a network, the Shortest Path First (SPF) is calculated for each configured topology.

The figure below depicts a non-congruent topology with IPv6 IS-IS MT enabled. Router 1 is an IPv4 and IPv6 dual stack router, Router 2 is an IPv6 router, and Router 3 is an IPv4 router. All the routers (Router 1, Router 2, and Router 3) in the Area 1 are configured with a set of independent topologies.

FIGURE 40 IS-IS non-congruent topology



Configuration considerations for IPv6 IS-IS MT

The following configuration considerations apply for IPv6 IS-IS MT

- The wide metric style must be configured before enabling IPv6 IS-IS MT.
- IPv4, IPv6, or IPv4 and IPv6 configured on the same interface must run on the same IS-IS level.
- Enabling or disabling IPv6 IS-IS MT clears all adjacencies, LSP databases, and IPv6 IS-IS routes.
- All routers on a point-to-point or a broadcast interface must support at least one common topology (IPv4 or IPv6), when MT is enabled.

Migrating to IPv6 IS-IS MT

The following steps must be performed to migrate from a non-MT environment to an MT environment.

1. Assume that the entire network is not an IPv6 IS-IS MT environment, and ensure that all the routes are correct.
2. Use the **multi-topology** command with the **transition** parameter to enable transition mode on each router one by one, and ensure that all the routes are correct.
3. After all the routers are in transition mode, use the **no multi-topology** command with the **transition** parameter to disable transition mode on each router one by one. Ensure that all the routes are correct.
4. Change the topology to make IPv4 and IPv6 different.

Maintaining MT adjacencies

With the extension of IPv6 IS-IS MT, the new type, length, and value (TLV) parameters are added into the IS to IS hello (IIH) packets that advertise the topologies of the interface. In IPv6 IS-IS MT, the router advertises its information using the new TLV parameters such as MT ID TLV, MT IS Reachability TLV, MT Reachable IPv4 TLV, and MT Reachable IPv6 TLV. The TLVs specify the types of data, the maximum length of the data, and the valid values for the data.

Forming adjacencies on the point-to-point interfaces

On a point-to-point interface, adjacencies are formed with IS-IS routers that do not implement MT extensions. If two peers share at least one common topology, then an adjacency is formed between the peers.

Forming adjacencies on the broadcast interfaces

On a broadcast interface, all the MT-enabled routers advertise their MT capability TLV in their IIH packets. The MT-enabled IS-IS routers form adjacency with any IS-IS routers whether or not MT is enabled. A peering MT-disabled IS-IS router does not form adjacency when NLPID TLVs do not match.

New TLV attributes

The new TLV parameters to support the IPv6 IS-IS MT extension are MT ID TLV, MT IS Reachability TLV, MT Reachable IPv4 TLV, and MT Reachable IPv6 TLV.

Enabling IPv6 IS-IS MT

You can enable IPv6 IS-IS MT in an area or a domain so that the MT-enabled devices runs IPv6 IS-IS in multi SPF mode. The following task enables IPv6 IS-IS MT with transition support.

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv6 unicast
   ```

4. Enter the **multi-topology** command with the **transition** parameter to enable IPv6 IS-IS MT with transition support.

   ```
   device(config-router-isis-ipv6u)# multi-topology transition
   ```

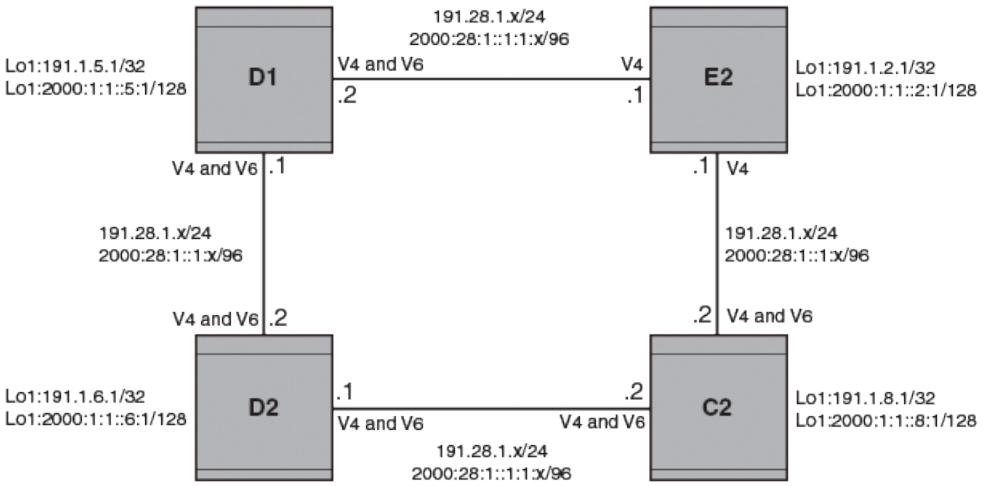
The following example enables IPv6 IS-IS MT with transition support.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# multi-topology transition
```

Configuration example to deploy IPv6 IS-IS MT

The figure below shows an example of a non-congruent topology enabled with IPv6 IS-IS MT. Device D1 supports both the IPv4 and IPv6 topologies, device D2 supports both the IPv4 and IPv6 topologies, device E2 supports an IPv4 topology, and device C2 supports both the IPv4 and IPv6 topologies.

FIGURE 41 IPv6 IS-IS MT configuration



Configuration commands to enable IPv6 IS-IS MT on device D1

The following commands enable IPv6 IS-IS MT on device D1.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# net 00.0000.001b.ed03.1400.00
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# metric-style wide
device(config-router-isis-ipv4u)# exit
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# multi-topology
device(config-router-isis-ipv6u)# exit
```

Configuration commands to enable IPv6 IS-IS MT on device D2

The following commands enable IPv6 IS-IS MT on device D2.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# net 00.0000.001b.ed04.4400.00
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# metric-style wide
device(config-router-isis-ipv4u)# exit
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# multi-topology
device(config-router-isis-ipv6u)# exit
```

Configuration commands to enable IPv6 IS-IS MT on device E2

The following commands enable IPv6 IS-IS MT on device E2.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# net 00.0000.001b.ed04.4000.00
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# metric-style wide
device(config-router-isis-ipv4u)# exit
device(config-isis-router)# address-family ipv6 unicast
```

```
device(config-router-isis-ipv6u)# multi-topology
device(config-router-isis-ipv6u)# exit
```

Configuration commands to enable IPv6 IS-IS MT on device C2

The following commands enable IPv6 IS-IS MT on device C2.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# net 00.0000.001b.ed04.0000.00
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# metric-style wide
device(config-router-isis-ipv4u)# exit
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# multi-topology
device(config-router-isis-ipv6u)# exit
```

Displaying IS-IS statistics

Various **show isis** commands verify information about IPv4 and IPv6 IS-IS configurations.

Use one or more of the following commands to verify IS-IS information. The commands do not have to be entered in this order.

1. Enter the **show isis** command.

```
device# show isis

IS-IS Routing Protocol Operation State: Enabled
IS-Type: Level-1-2
System Id: 768e.f805.5812
Manual area address(es): 11
Level-1-2 Database State: On
Administrative Distance 115
Maximum Paths 8
Default redistribution metric 0
Default link metric for level-1 0 (conf)/ 10 (adv)
Default link metric for level-2 0 (conf)/ 10 (adv)
Protocol Routes Redistributed into IS-IS: None
Number of Routes Redistributed into IS-IS: 0
Level-1 Auth-mode: None
Level-2 Auth-mode: None
Metric Style Supported for Level-1: Narrow
Metric Style Supported for Level-2: Narrow
Graceful-Restart Helper Support: Enabled
ISIS Partial SPF Optimizations: Enabled
Timers:
L1 SPF: Max-wait 5s Init-wait 5000ms Second-wait 5000ms
L2 SPF: Max-wait 5s Init-wait 5000ms Second-wait 5000ms
L1 SPF is not scheduled
L2 SPF is not scheduled
PSPF: Max-wait 5000ms Init-wait 2000ms Second-wait 5000ms
PSPF is not scheduled
LSP: max-lifetime 1200s refresh-interval 900s gen-interval 10s
retransmit-interval 5s, lsp-interval 33ms
SNP: csnp-interval 10s psnp-interval 2s
Global Hello Padding: Enabled
Global Hello Padding For Point to Point Circuits: Enabled
Ptpt Three Way HandShake Mechanism: Enabled
BGP Ipv4 Converged: False BGP Ipv6 Converged: False
IS-IS Traffic Engineering Support: Disabled
No ISIS Shortcuts Configured
BFD: Disabled, BFD HoldoverInterval: 0
NSR: Disabled
LSP-SYNC: Not Globally Enabled
```

   This example output displays general IS-IS information..

2. Enter the **show isis config** command.

```
device# show isis config

router isis
net 11.768e.f805.5812.00
address-family ipv4 unicast
!
address-family ipv6 unicast
!
```

   This example shows the global IS-IS configuration commands that are in effect on the device..

3. Enter the **show isis counts** command.

```
device# show isis counts

Area Mismatch: 0
Max Area Mismatch: 0
System ID Length Mismatch: 0
LSP Sequence Number Skipped: 0
LSP Max Sequence Number Exceeded: 0
Level-1 Database Overload: 1
Level-2 Database Overload: 0
Our LSP Purged: 2
```

This example shows IS-IS error statistics..

4. Enter the **show isis hostname** command.

```
device# show isis hostname

Total number of entries in IS-IS Hostname Table: 1
System ID Hostname * = local IS
--------------------------------------------------
* 768e.f805.5812 R1
```

This example shows the router-name-to-system-ID mapping table entries for the device.

5. Enter the **show isis interface** command.

```
device# show isis interface

Total number of IS-IS Interfaces: 11
Interface: Ve 301
Circuit State: UP Circuit Mode: Level 1-2
Circuit Type: BCAST Passive State: FALSE
Circuit Number: 2, MTU: 1500
Level-1 Auth-mode: NONE
Level-2 Auth-mode: NONE
Level-1 Metric: 10, Level-1 Priority: 64
Level-1 Hello Interval: 10, Level-1 Hello Multiplier: 3
Level-1 Designated IS: R1-02 Level-1 DIS Changes: 2
Level-2 Metric: 10, Level-2 Priority: 64
Level-2 Hello Interval: 10, Level-2 Hello Multiplier: 3
Level-2 Designated IS: R1-02 Level-2 DIS Changes: 2
Next IS-IS LAN Level-2 Hello in 11 seconds
Number of active Level-2 adjacencies: 0
Next IS-IS LAN Level-1 Hello in 1 seconds
Number of active Level-1 adjacencies: 0
Circuit State Changes: 1 Circuit Adjacencies State Changes: 0
Rejected Adjacencies: 0
Circuit Authentication L1 failures: 0
Circuit Authentication L2 failures: 0
Bad LSPs: 0
Control Messages Sent: 7577 Control Messages Received: 0
Hello Padding: Enabled
IP Enabled: TRUE
IP Addresses:
11.2.1.1/30
IPv6 Enabled: FALSE
MPLS TE Enabled: FALSE
BFD Enabled: FALSE
LDP-SYNC: Disabled, State:
…
```

This example shows information about IS-IS interfaces for a device.

6. Enter the **show isis interface** command, using the **brief** keyword.

```
device# show isis interface brief

Total number of IS-IS Interfaces: 11
Interface Type State Mode Passive MTU UpAdj DIS StateChg AdjStateChg
Ve 301 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 302 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 303 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 304 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 305 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 306 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 307 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 308 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 309 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 310 BCAST UP L12 FALSE 1500 0 None 1 0
Lo 1 BCAST UP L12 TRUE 0 0 None 1 0
```

This example shows summarized information about IS-IS interfaces for a device.

7. Enter the **show isis neighbor** command.

```
device# show isis neighbor
```

This example shows IS-IS neighbor information.

8. Enter the **show isis spf-log** command.

```
device# show isis spf-log

ISIS Level-1 SPF Log
When Duration Nodes Count Last-Trigger-LSP Trigger
10h34m13s 0ms 1 10 R1.00-00 Interface State Change
10h34m38s 0ms 1 2 R1.00-00 Interface Config Change
10h34m43s 0ms 1 18 R1.00-00 Interface Config Change
10h34m48s 0ms 1 5 R1.00-00 Interface State Change
ISIS Level-2 SPF Log
When Duration Nodes Count Last-Trigger-LSP Trigger
10h34m13s 0ms 1 10 R1.00-00 Interface State Change
10h34m38s 0ms 1 2 R1.00-00 Interface Config Change
10h34m43s 0ms 1 18 R1.00-00 Interface Config Change
10h34m48s 0ms 1 5 R1.00-00 Interface State Change
```

This example shows IS-IS link-state packet (LSP) logging information.

9. Enter the **show isis traffic** command.

```
show isis traffic

                            Message Received     Message Sent
 Level-1 Hellos             0                    44912
 Level-2 Hellos             0                    44927
 PTP Hellos                 0                    0
 Level-1 LSP                0                    0
 Level-2 LSP                0                    0
 Level-1 CSNP               0                    0
 Level-2 CSNP               0                    0
 Level-1 PSNP               0                    0
 Level-2 PSNP               0                    0
```

This example shows information about IS-IS packet counts..

OSPFv2

- OSPFv2 overview 319
- Autonomous System 320
- OSPFv2 components and roles 320
- Enabling OSPFv2 322
- Backbone area 322
- Assigning OSPFv2 areas 323
- Area range 323
- Area types 324
- Stub area and totally stubby area 324
- Not-so-stubby area (NSSA) 325
- Assigning interfaces to an area 327
- Link state advertisements 328
- Configuring an MD5 password and authentication change hold time for an OSPFv2 interface 328
- Virtual links 328
- Default route origination 330
- External route summarization 331
- Modifying Shortest Path First timers 331
- OSPFv2 administrative distance 332
- OSPFv2 LSA refreshes 332
- OSPFv2 graceful restart 333
- OSPFv2 non-stop routing 335
- Redistributing routes into OSPFv2 336
- OSPFv2 type 3 LSA filtering 336
- OSPFv2 over VRF 339
- Configuring the OSPFv2 Max-Metric Router LSA 339
- Re-enabling OSPFv2 compatibility with RFC 1583 340
- Changing default settings 340
- Disabling and re-enabling OSPFv2 event logging 340
- Understanding the effects of disabling OSPFv2 341
- Displaying OSPFv2 results 342

OSPFv2 overview

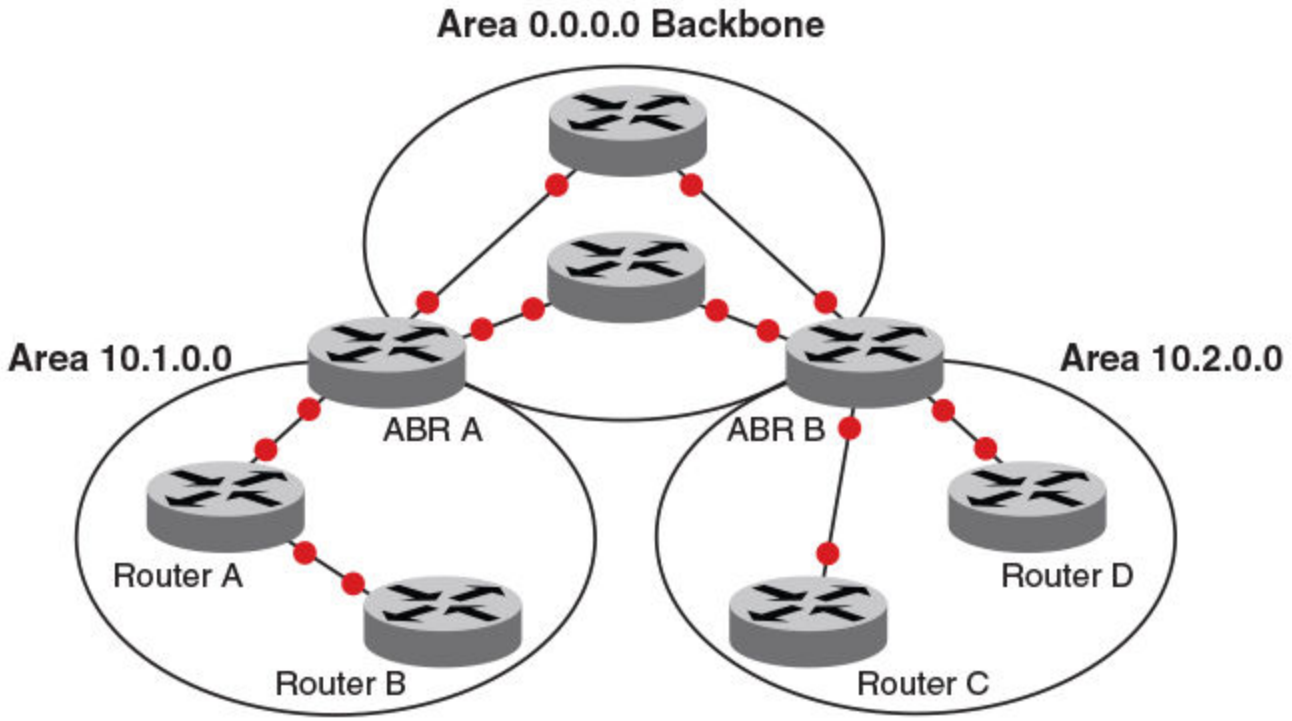
Open Shortest Path First Version 2 (OSPFv2) is a link-state routing protocol that uses link-state advertisements (LSAs) to update neighboring routers about a router's interfaces. Each router maintains an identical area-topology database to determine the shortest path to any neighboring router.

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 42 OSPF operating in a network



NOTE
For details of components and virtual links, refer to OSPFv2 components and roles on page 320 and Virtual links on page 328, 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

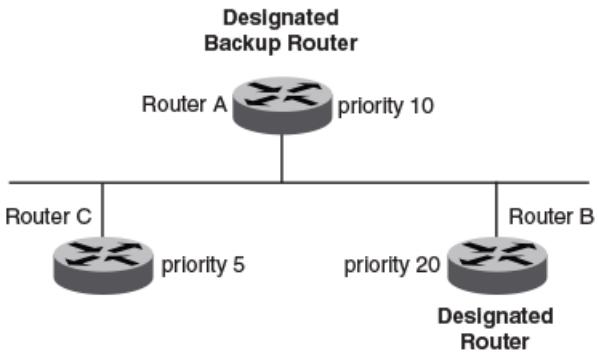
In an OSPF broadcast network, OSPF elects one router to serve as the designated router (DR) and another router on the segment to act as the backup designated router (BDR). This minimizes the amount of repetitive information that is forwarded on the network. OSPF forwards all messages to the designated 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.

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 43 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.

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. Assign the areas to which the device will be attached.
3. Assign individual interfaces to the OSPF areas.
4. Assign a virtual link to any ABR that does not have a direct link to the OSPF backbone area.
5. Refer to Changing default settings on page 340.

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.

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-router-ospf-vrf-default-vrf)# area 0
```

4. Enter the **area** command to define a second OSPFv2 area ID.

```
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-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 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 **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

3. 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 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 **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

3. 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-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-router-ospf-vrf-default-vrf)# area 40 stub 99 no-summary
```

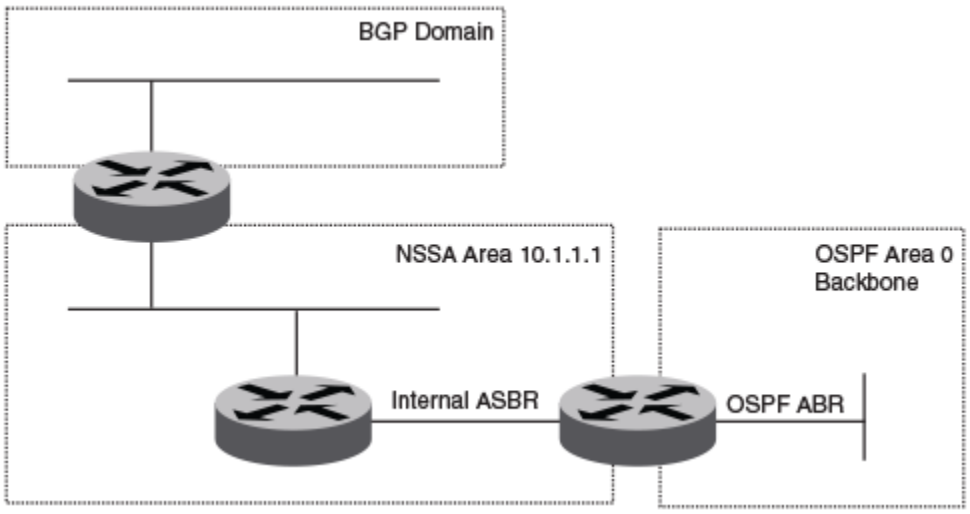
Not-so-stubby area (NSSA)

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 44 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.

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 **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config)# router ospf
   ```

3. Enter the **area nssa** command and specify an area address and a cost.

   ```
   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(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 **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

3. Enter the **area nssa** command, specifying an area and a cost.

```
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 nssa 10
```

4. Enter the **summary-address** command, followed by the IP address and mask for the summary route.

```
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-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
```

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-Loopback-2)# ip ospf area 10.5.0.0
```

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-Loopback-2)# ip ospf area 10.5.0.0
```

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.
- 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
```

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
```

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
```

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
```

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:

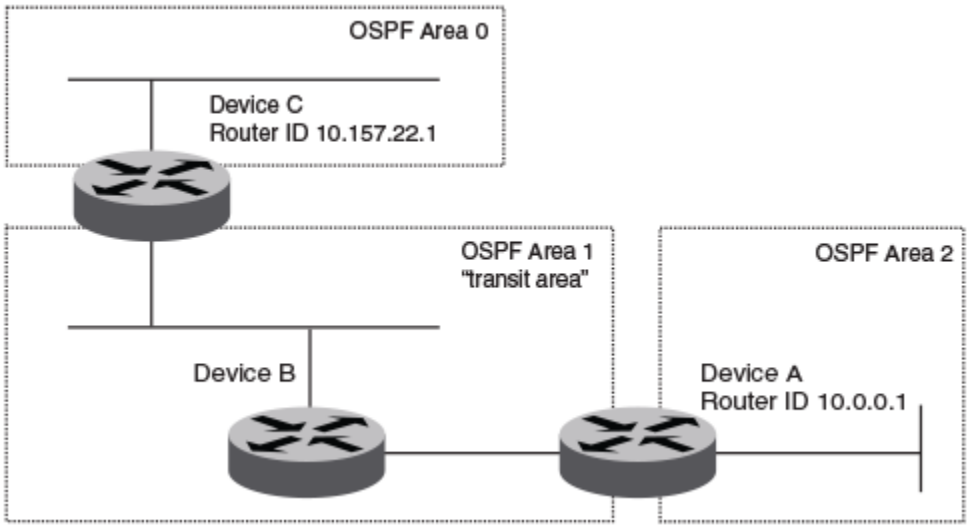
- The transit area ID represents the shared area of the two ABRs and serves as the connection point between the two routers. This number should match the area ID value.
- The neighbor router field is the router ID (IP address) of the router that is physically connected to the backbone, when assigned 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 45 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 **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 assign an OSPFv2 area ID.

```
device(config-router-ospf-vrf-default-vrf)# area 0
```

4. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-router-ospf-vrf-default-vrf)# area 1
```

5. 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-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.2.2.2
```

6. On ABR2, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

7. Enter the **router ospf** command to enter OSPFv2 router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

8. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-router-ospf-vrf-default-vrf)# area 1
```

9. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-router-ospf-vrf-default-vrf)# area 2
```

10. 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-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-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
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.

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 **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config)# router ospf
```

3. Enter the **timers** command with the **throttle spf** keyword and specify the SPF delay, the hold time, and the maximum wait time.

```
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-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.

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 **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

   ```
   device(config)# router ospf
   ```

3. Enter the **timers** command with the **lsa-group-pacing** parameter.

   ```
   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-router-ospf-vrf-default-vrf)# timers lsa-group-pacing 120
```

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).

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 **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config)# router ospf
```

3. Enter the **no graceful restart** command to disable GR on the device.

```
device(config-router-ospf-vrf-default-vrf)# no graceful-restart
```

The following example disables GR.

```
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 **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config)# router ospf
```

3. Enter the **graceful restart** command to re-enable GR on the device.

```
device(config-router-ospf-vrf-default-vrf)# graceful-restart
```

4. 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-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-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 **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

3. Enter the **graceful-restart** command using the **helper-disable** keyword to disable the GR helper.

```
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-router-ospf-vrf-default-vrf)# graceful-restart helper-disable
```

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 virtual links, so traffic loss is expected while performing hitless failover.

NOTE
NSR and Graceful Restart (GR) are mutually exclusive.

NOTE
Process restart takes precedence over GR or non-stop routing (NSR).

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 **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config)# router ospf
```

3. Enter the **nonstop-routing** command to re-enable NSR on the device.

```
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-router-ospf-vrf-default-vrf)# nonstop-routing
```

Redistributing routes into OSPFv2

OSPFv2 routes can be redistributed, and the routes to be redistributed can be specified.

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-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 **bgp** parameter to redistribute BGP routes.

```
device(config-router-ospf-vrf-default-vrf)# redistribute bgp
```

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
```

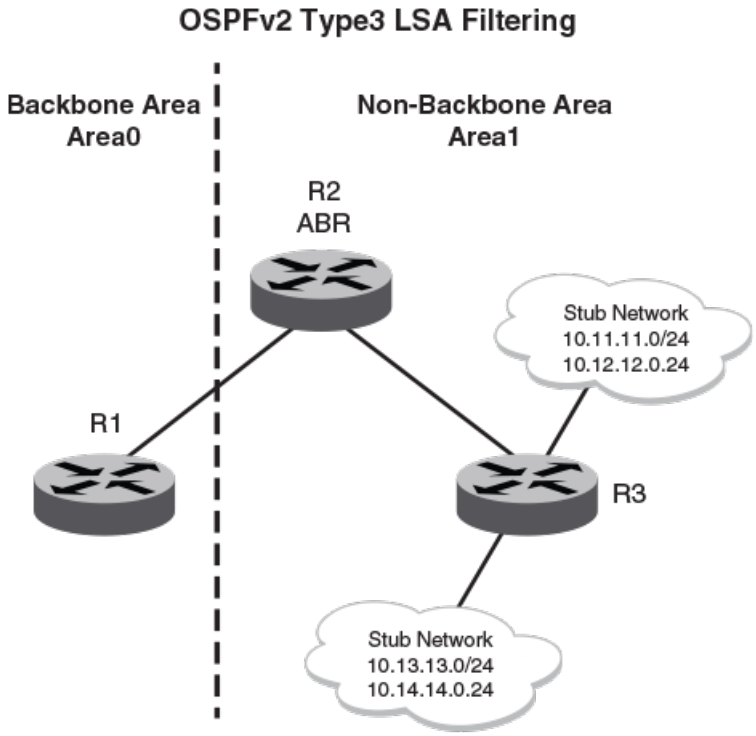
OSPFv2 type 3 LSA filtering

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 46 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 32 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 **ip prefix-list** command and specify a name.

   ```
   device(config)# 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.

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config)# router ospf
   ```

4. Enter the **area** command to define an OSPFv2 area ID.

   ```
   device(config-router-ospf-vrf-default-vrf)# area 1
   ```

5. Enter the **area prefix-list** command and specify an area address and a prefix-list.

   ```
   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-router-ospf-vrf-default-vrf)# area 1
device(config-router-ospf-vrf-default-vrf)# area 1 prefix-list mylist out
```

OSPFv2 over VRF

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 **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)# 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-router-ospf-vrf-green)#
```

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 **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

3. 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 the maximum metric value using the **all-lsas** parameter.

```
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 **ip router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

3. 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)# router ospf
device(config-router-ospf-vrf-default-vrf)# rfc1583-compatibility
```

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 **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config)# router ospf
```

3. Enter the **no log all** command to disable the logging of all OSPFv2 events.

```
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-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 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.

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 **no router ospf** command to disable OSPFv2 on the device.

```
device(config)# no router ospf
```

The following example disables OSPFv2 on a device.

```
device# configure terminal
device(config)# 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 33 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 34 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)

OSPFv3

- OSPFv3 overview 347
- Configuring the router ID 348
- Enabling OSPFv3 348
- Configuring OSPFv3 348
- OSPFv3 areas 349
- Virtual links 354
- OSPFv3 route redistribution 357
- Default route origination 359
- Disabling and re-enabling OSPFv3 event logging 359
- Filtering OSPFv3 routes 360
- SPF timers 360
- OSPFv3 administrative distance 361
- Changing the reference bandwidth for the cost on OSPFv3 interfaces 362
- OSPFv3 LSA refreshes 363
- External route summarization 363
- OSPFv3 over VRF 364
- Setting all OSPFv3 interfaces to the passive state 365
- OSPFv3 graceful restart helper 365
- OSPFv3 non-stop routing 366
- OSPFv3 max-metric router LSA 367
- IPsec for OSPFv3 368
- Displaying OSPFv3 results 372

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.

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 **ip router-id** command to specify the router ID.

```
device(config) 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
```

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.

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 10.11.12.13
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# 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-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.

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 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 **ip router-id** command to specify the router ID.

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

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config)# ipv6 router ospf
   ```

4. Enter the **area** command to define an OSPFv3 area ID.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# area 0
   ```

5. Enter the **area** command to define a second OSPFv3 area ID.

   ```
   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-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
```

3. Enter the **ipv6 address** command to add an IPv6 address to the interface.

```
device(conf-if-eth-1/1)# ipv6 address 2001:1:0::1:1/64
```

4. Enter the **ipv6 ospf area** command.

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

6. Enter the **interface** command and specify an interface.

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

7. Enter the **ipv6 address** command to add an IPv6 address to the interface.

```
device(conf-if-eth-2/1)# ipv6 address 2001:1:0::2:1/64
```

8. Enter the **ipv6 ospf area** command.

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

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 **ip router-id** command to specify the router ID.

   ```
   device(config) ip router-id 10.4.4.4
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config)# ipv6 router ospf
   ```

4. Enter the **area stub** command and specify a metric value.

   ```
   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-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.

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 **ip router-id** command to specify the router ID.

   ```
   device(config) ip router-id 10.3.3.3
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config)# ipv6 router ospf
   ```

4. Enter the **area nssa** command with the **default-information-originate** keyword and specify a cost.

   ```
   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-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.

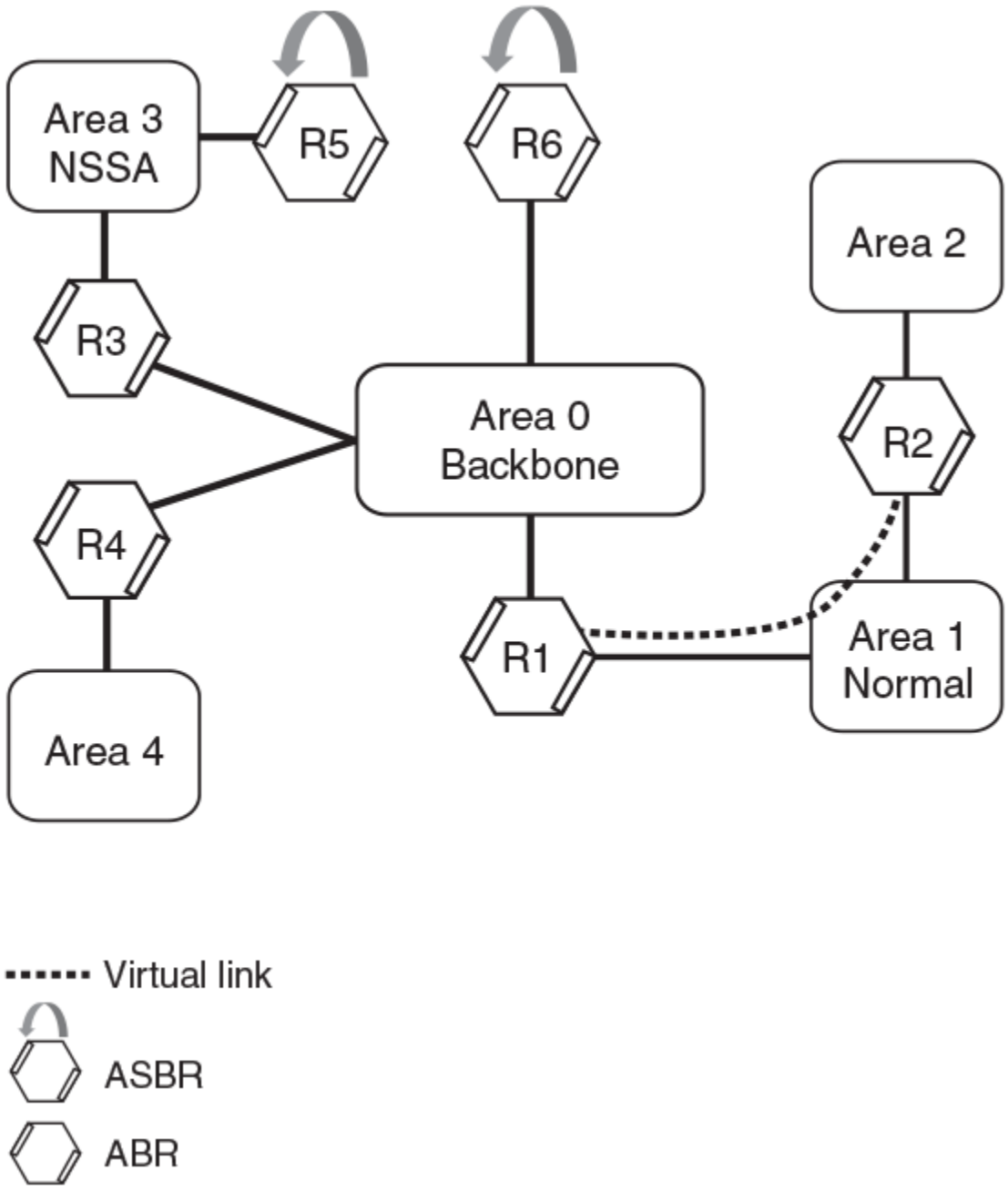
Virtual links

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 47 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 **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 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

4. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 0
```

5. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1
```

6. 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-ipv6-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 **ip router-id** command to specify the router ID.

```
device(config) ip router-id 10.2.2.2
```

9. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

10. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1
```

11. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 2
```

12. 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-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-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

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.

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 **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

3. Enter the **redistribute** command with the **static** parameter to redistribute static routes.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute static
```

4. Enter the **redistribute** command with the **bgp** parameter to redistribute static routes.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute bgp
```

5. On device2, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

6. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

7. Enter the **redistribute** command with the **connected** and **route-map** parameters to redistribute connected routes and specify a route map.

```
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-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-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 **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   ```

3. Enter the **default-information-originate** command with the **always**, **metric**, and **metric-type** parameters.

   ```
   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-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 configuration mode and enable OSPFv3 globally.

```
device(config)# ipv6 router ospf
```

3. 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)# 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.

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 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 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 **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config)# ipv6 router ospf
```

3. Enter the **timers** command with the **spf** parameter.

```
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-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 **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 **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 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 **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config)# ipv6 router ospf
```

3. 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
```

The following example changes the auto-cost reference bandwidth to 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 **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   ```

3. Enter the **timers** command with the **lsa-group-pacing** parameter.

   ```
   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-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.

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.

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)#
```

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 **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config)# ipv6 router ospf
```

3. Enter the **default-passive-interface** command to mark all interfaces passive by default.

```
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 **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config)# ipv6 router ospf
```

3. 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.

```
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 **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config)# ipv6 router ospf
```

3. 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-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-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 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 **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config)# ipv6 router ospf
```

3. Enter the **graceful restart** command to re-enable GR on the device.

```
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-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 **ip router-id** command to specify the router ID.

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

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

4. Enter the **max-metric router-lsa** command with the **external-lsa** keyword and specify a value to configure the maximum metric value .

```
device(config-ipv6-router-ospf-vrf-default-vrf)# max-metric router-lsa external-lsa 1500
```

5. 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-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-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 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.

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.

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, 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
- Key rollover timer
- Specifying the key add remove timer

Configuring IPsec on an OSPFv3 area

IPsec can be configured to secure communications on an OSPFv3 area.

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 **ip router-id** command to specify the router ID.

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

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 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 AH and MD5 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-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.

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
```

3. Enter the **ipv6 ospf area** command to assign a specified area to the interface.

```
device(conf-if-eth-1/1)# ipv6 ospf area 0
```

4. Enter **ipv6 ospf authentication spi** *value* **esp null hmac-sha1** and specify a 40-character hexadecimal key.

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

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 **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 router.

   ```
   device(config)# ipv6 router ospf
   ```

4. 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)# 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 **ip router-id** command to specify the router ID.

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

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config)# ipv6 router ospf
   ```

4. 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-ipv6-router-ospf-vrf-default-vrf)# key-add-remove-interval 240
   ```

5. Enter the **key-rollover-interval** command and specify the desired interval to set the timing of the configuration changeover.

```
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-ipv6-router-ospf-vrf-default-vrf)# key-add-remove-interval 240
device(config-ipv6-router-ospf-vrf-default-vrf)# key-rollover-interval 240
```

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 35 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 36 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)

Multi-VRF

- Multi-VRF overview....379
- Configuring Multi-VRF....381
- Multi-VRF configuration example....384
- Inter-VRF route leaking....390
- Internet route scaling overview....399

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.

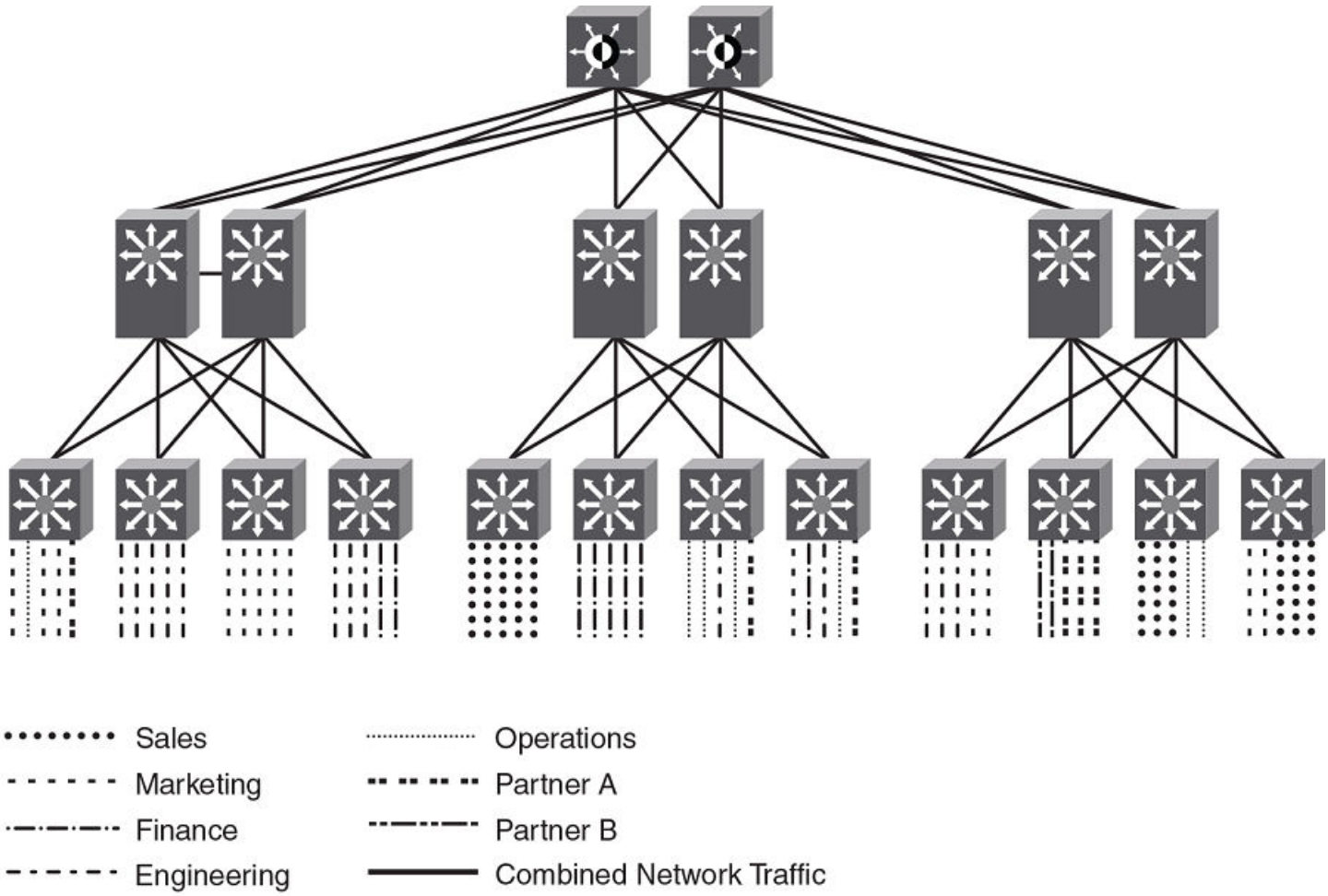
Central to VRF-Lite is the ability to maintain multiple VRF tables on the same Provider Edge (PE) Router. VRF-Lite uses multiple instances of a routing protocol such as OSPF or BGP to exchange route information for a VPN among peer PE routers. The VRF-Lite capable PE router maps an input customer interface 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 48 Example high-level Multi-VRF topology



A Multi-VRF instance can be configured on any of the following:

- Virtual interfaces
- Loopback interfaces
- Ethernet interfaces

To configure Multi-VRF, perform the following steps:

- Configure 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.

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 37 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.)

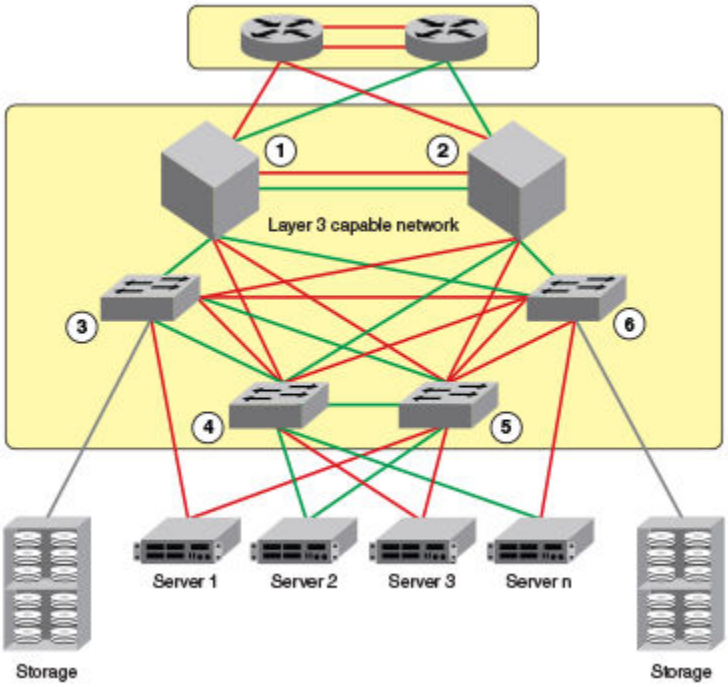
   ```
   device(vrf-myvrf-ipv4-unicast)# max-route 3600
   ```

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 49 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 38 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	0/1

TABLE 38 Topology details (continued)

Node	Description	Networks	Carries routes . . .	Interfaces
			10.2.3.0/24	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. This is useful in cases where multiple VRFs share the same path to reach 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 a single server.

Both static and dynamic route leaking are supported. Each routed interface (whether 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. The user 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.

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

> ATTENTION
> This feature should be deployed only by an advanced user, as route leak configuration in source VRFs may collide with route/interface definitions in target VRFs. This may lead to unpredictable behavior in packet forwarding.

Some of the ways that leaked route conflicts can occur are the following:

- Static route conflict
- Dynamic route conflict
- Connected route conflict

A static route conflict may occur when the same prefix is reachable by two different next hops in the target VRF. The forwarding behavior would be different depending on which command occurred later. This following example global configuration lines would present 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 mentioned above (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/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.

> ATTENTION
> 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.

A connected route conflict is illustrated by the following example configuration lines:

```
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

The user must be aware of such possible conflicts before deploying the route leak feature, as currently there is no error checking for these scenarios. A good rule is to make sure 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.

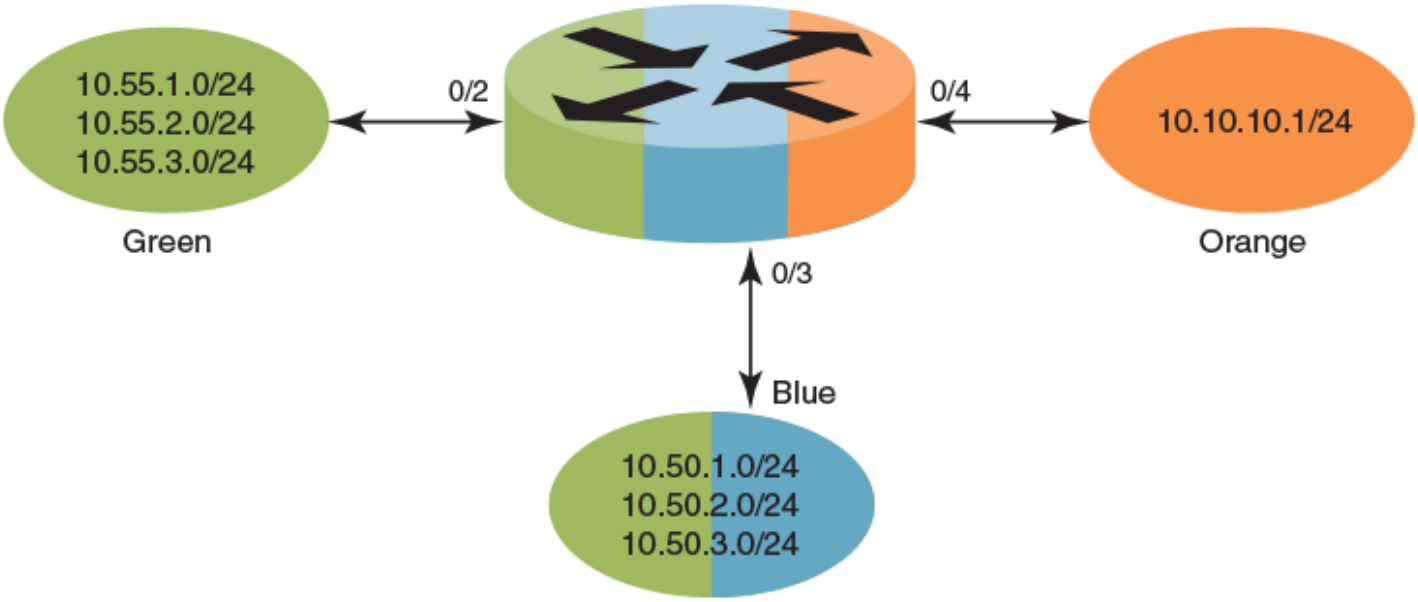
1. Create the VRF instances you want to be the leaker (source VRF) and where the route is being leaked to (destination VRF).
2. Specify the interface for the source VRF and map it to the source VRF.
3. Enter the IP address/mask to be used for this VRF instance.
4. Specify the interface you want to be 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 50 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
```

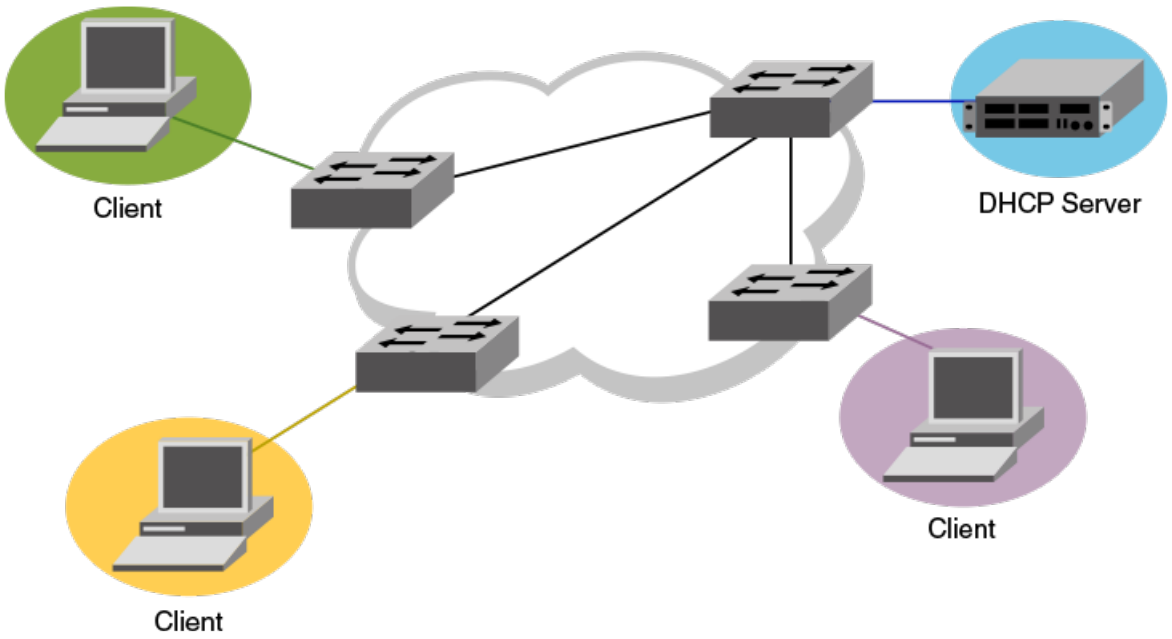
8. (Optional) For bidirectional route-leak traffic, you can also configure route leaking from VRF "Green" to VRF "Blue".

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 51 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.

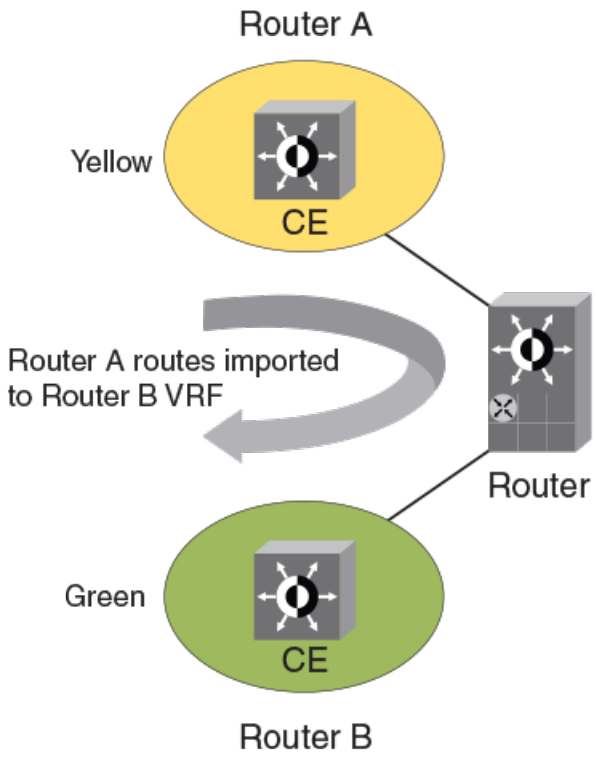
1. Configure the VRF instances you want to be the leaker (source VRF) and where the route is being leaked to (destination VRF).
2. Specify the interface for the source VRF and map it to the source VRF.
3. Enter the IP address/mask to use for this VRF instance.
4. Specify the interface you want to be 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 52 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 39 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 overview

Internet route scaling, also known as a optiscale is most commonly used for routing. This forwards packet received from the input interface.

Internet route scaling allows the software to program dual stack internet routes into the hardware. Internet route scaling has two parts:

- FIB compression
- Hardware optimization

Internet route scaling also known as optiscale limitations are:

- uRPF cannot coexist
- Only support in default VRF

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 401
- Enabling a master VRRP device 406
- Enabling a backup VRRP device 408
- VRRP multigroup clusters 408
- Tracked ports and track priority with VRRP and VRRP-E 411
- VRRP backup preemption 413
- Accept mode for backup VRRP devices 414
- Virtual router MAC address 414
- VRRP-Ev2 overview 416
- Enabling a VRRP-E device 417
- Configuring MD5 authentication on IPv4 VRRP-E interfaces 418
- Track routes and track priority with VRRP-E 419
- VRRP-E load-balancing using short-path forwarding 421
- Displaying VRRPv2 information 423
- Clearing VRRPv2 statistics 425

VRRPv2 overview

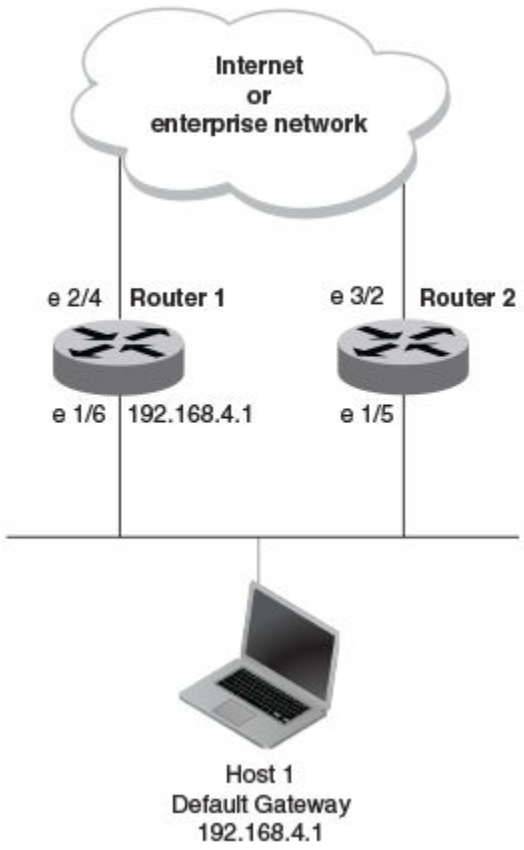
Virtual Router Redundancy Protocol (VRRP) is an election protocol that provides redundancy to routers within a Local Area Network (LAN).

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.

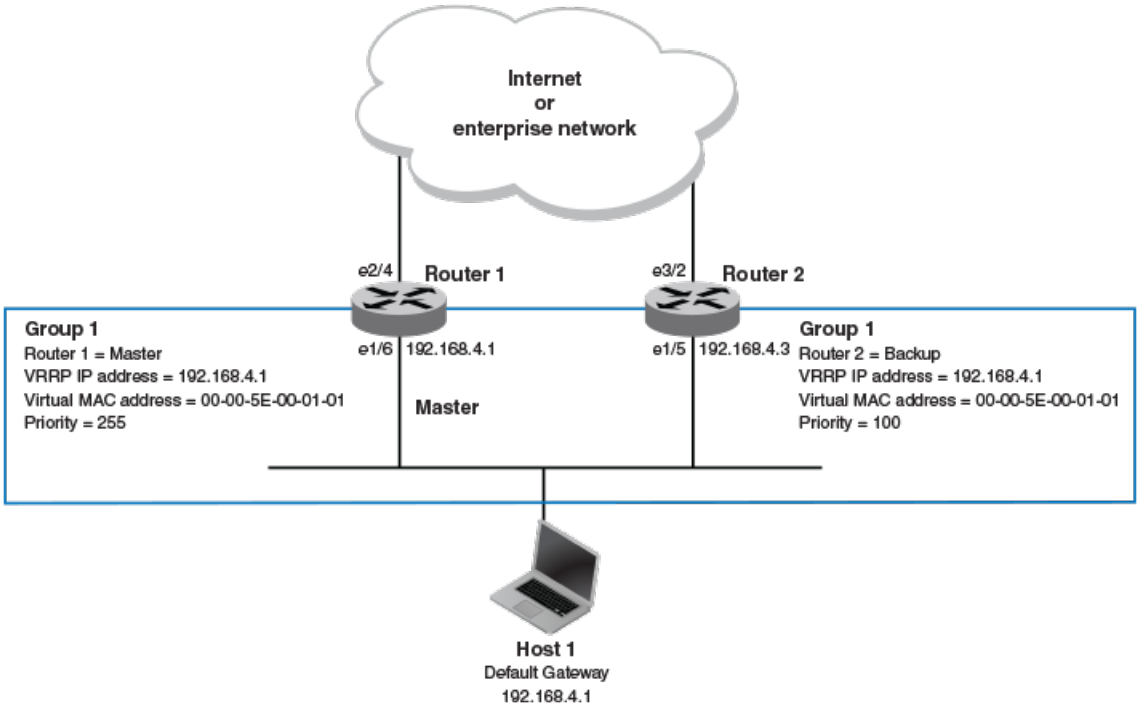
FIGURE 53 Single point of failure with Device 1 being the Host1 default gateway



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 54 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, OSPF, and ISIS. Their operation is unaffected when VRRP is enabled on the same interface as BGP4, OSPF, or ISIS.

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 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.

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 40 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.

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.
- 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.

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

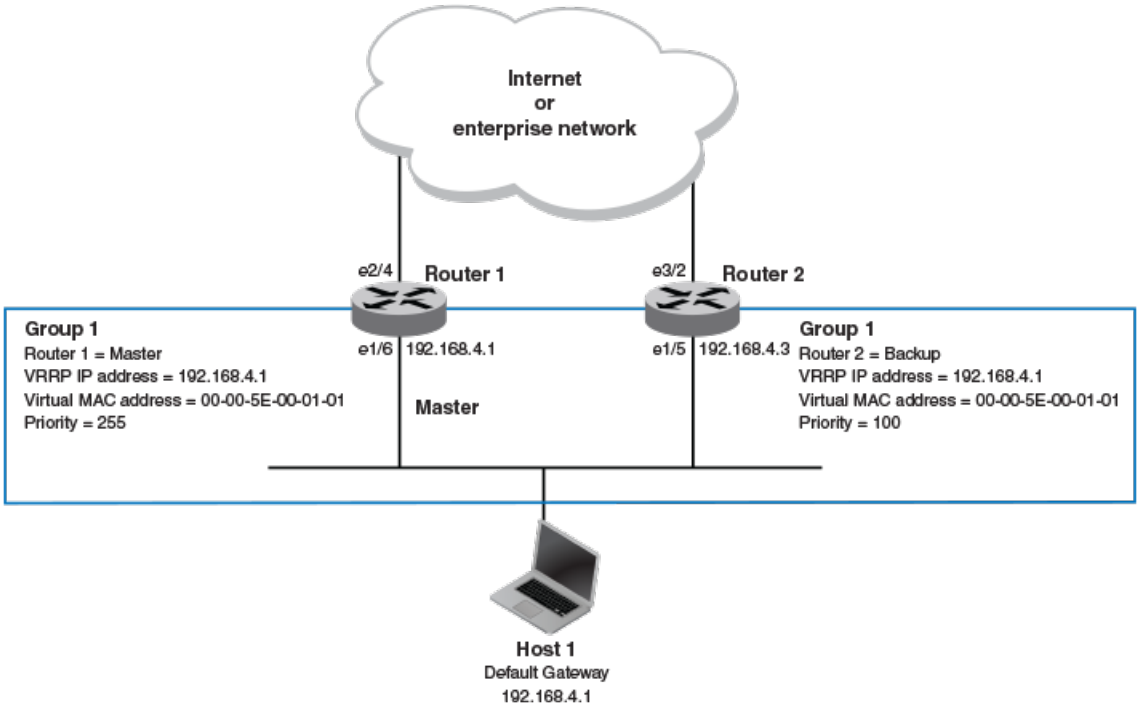
When a VRRP device (either master or backup) sends an ARP request or a reply packet, the MAC address of the sender is the MAC address of the router interface. One exception is if the owner sends an ARP request or a reply packet, in which case the MAC address of the sender is the virtual MAC address. 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 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 55 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. Globally enable VRRP.

```
device(config)# protocol 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(conf-if-eth-1/6)# ip address 192.168.4.1/24
```

5. Assign Router 1 to a group called Group 1.

```
device(conf-if-eth-1/6)# 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.

The following example configures a VRRP master device.

```
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
```

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 55 on page 407 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. Globally enable VRRP.

   ```
   device(config)# protocol vrrp
   ```

3. Configure the Ethernet interface for Router 2.

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

4. Configure the IP address of interface:

   ```
   device(config-if-eth-1/5)# ip address 192.168.4.3/24
   ```

   NOTE
   This router will become the backup router to Router 1.

5. Assign Router 2 to the same VRRP group as Router 1.

   ```
   device(config-if-eth-1/5)# vrrp-group 1
   ```

6. 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.

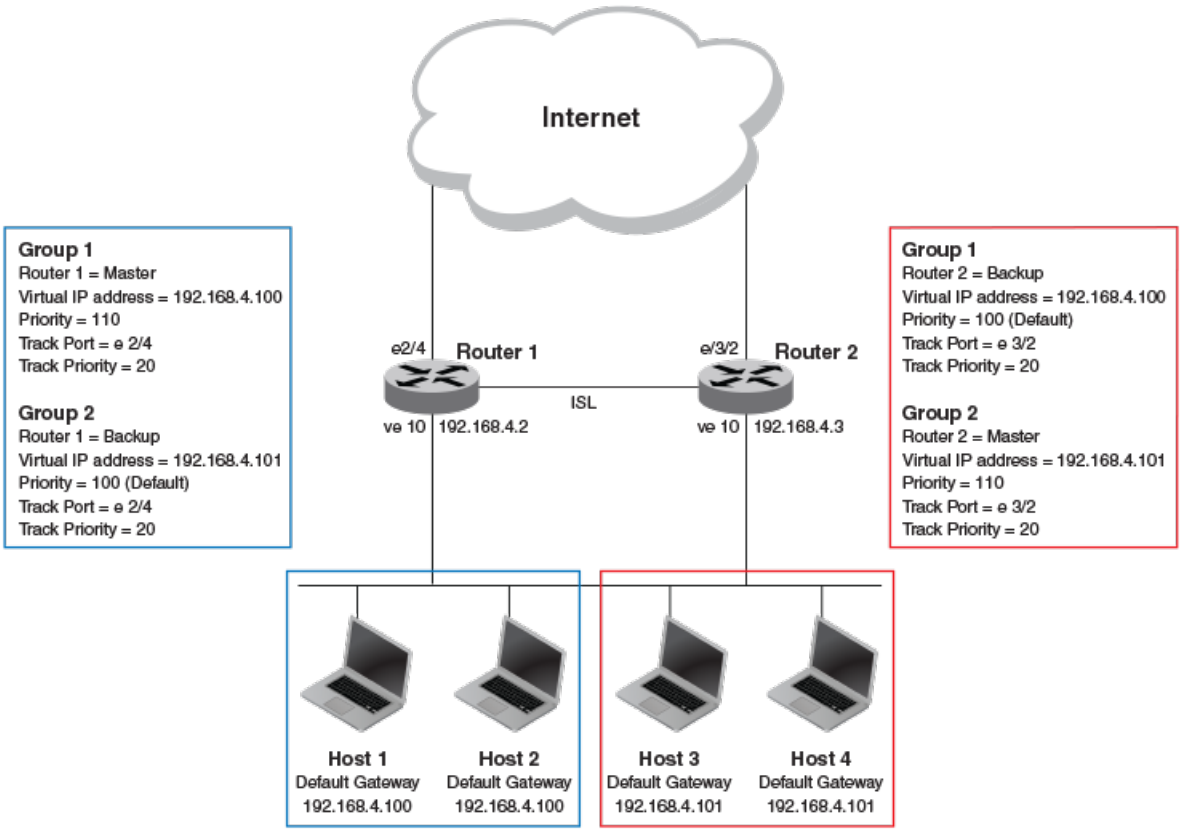
```
device# configure
device(config)# protocol vrrp
device(config)# interface ethernet 1/5
device(config-if-eth-1/5)# ip address 192.168.4.3/24
device(config-if-eth-1/5)# vrrp-group 1
device(config-vrrp-group-1)# virtual-ip 192.168.4.1
```

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 56 Two routers configured for dual redundant network access for the host



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 56 on page 409, 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. Enable the VRRP-E protocol globally.

```
device(config)# protocol vrrp-extended
```

2. Create a VLAN.

```
device(config)# vlan 10
```

3. Bind the VLAN to the virtual Ethernet (ve) interface link for Router 1.

```
device(config-vlan 10)# router-interface ve 10
```

4. Enter virtual Ethernet (ve) interface 10.

```
device(config-vlan 10)# interface ve 10
```

5. Configure the IP address of the ve link for Router 1.

```
device(config-if-Ve-10)# ip address 192.168.4.2/24
```

6. To assign Router 1 to a VRRP-E group called Group 1, enter the command:

```
device(config-if-Ve-10)# vrrp-extended-group 1
```

7. 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
```

8. 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.

9. 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)# 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
```

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)# 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
```

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)# 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
```

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:

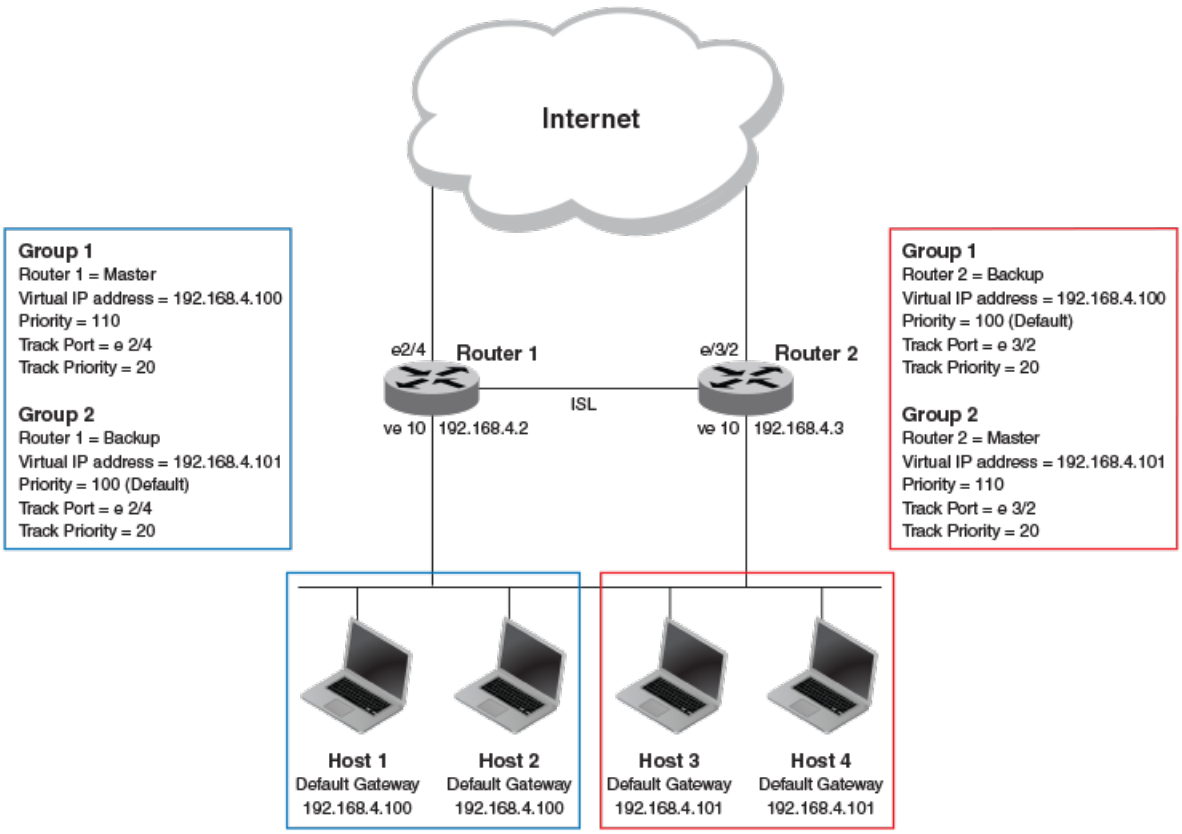
- Track priorities must be lower than VRRP or VRRP-E priorities.
- The dynamic change of router priority can trigger a master device switchover if preemption is enabled. However, if the router is an owner, the master device switchover will not occur.
- The maximum number of interfaces that can be tracked 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 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.

FIGURE 57 Multigroup VRRP routing topology



1. Enable VRRP globally.

```
device(config)# protocol vrrp
```

2. Enter interface configuration mode and run the following command:

```
device(config)# interface ve 10
```

3. Run the following command to enter group configuration mode.

```
device(config-if-Ve-10)# vrrp-group 1
```

4. Enter the **track** command to set the track port and priority:

```
device(config-vrrp-group-1)# track ethernet 2/4 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)# 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
```

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 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. From global configuration mode, configure the ethernet interface for the device.

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

2. Configure the IP address of the interface:

   ```
   device(conf-if-eth-1/5)# ip address 192.168.4.3/24
   ```

   NOTE
   This router is a backup router.

3. Assign the device to VRRP group 1.

   ```
   device(conf-if-eth-1/5)# vrrp-group 1
   ```

4. 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)# 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
```

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.

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.

   ```
   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.

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 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
```

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. 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. Configure the IP address of the interface.

   ```
   device(config-if-Ve-10)# ip address 192.168.4.1/24
   ```

5. Assign the device to a group called Group 1.

   ```
   device(config-if-Ve-10)# vrrp-extended-group 1
   ```

6. 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.

7. 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)# 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)# 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
```

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. Enable VRRP-E globally.

```
device(config)# protocol vrrp-extended
```

2. Enter interface configuration mode.

```
device(config)# interface ve 100
```

3. Run the following command to enter group configuration mode.

```
device(config-if-Ve-100)# vrrp-extended-group 1
```

4. 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
```

5. Return to privileged EXEC mode.

```
device(config-vrrp-group-1)# end
```

6. To view tracked networks with their priority and status, enter the following command:

```
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)# 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
```

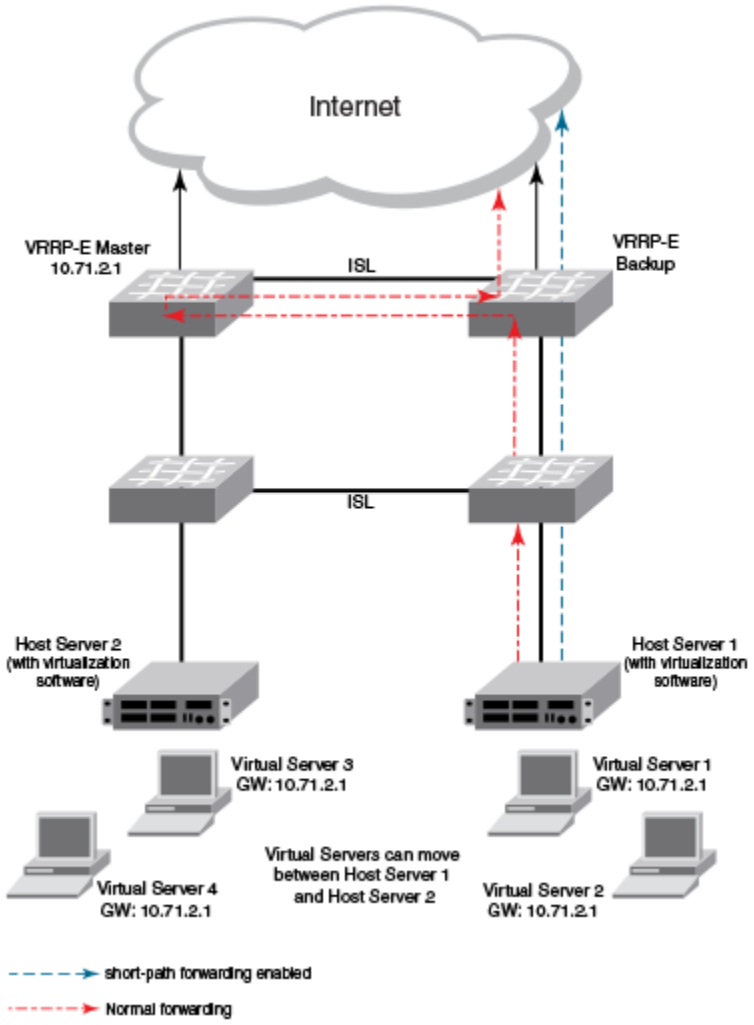
VRRP-E load-balancing using short-path forwarding

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.

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 58 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. Globally enable VRRP-E.

   ```
   device(config)# protocol vrrp-extended
   ```

3. Enter the **interface ve** command with an associated VLAN number.

   ```
   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.

4. Enter an IP address for the interface using the **ip address** command.

   ```
   device(config-ve-2019)# ip address 192.168.4.1/24
   ```

5. 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.

6. 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)# 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
```

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

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 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.

3. 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
```

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

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 **clear vrrp statistics** command.

```
device# clear vrrp statistics
```

4. 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.

VRRPv3

- VRRPv3 overview 427
- Enabling IPv6 VRRPv3 428
- Enabling IPv4 VRRPv3 429
- Tracked ports and track priority with VRRP and VRRP-E 430
- VRRP hold timer 431
- Accept mode for backup VRRP devices 433
- Alternate VRRPv2 checksum for VRRPv3 IPv4 sessions 434
- VRRPv3 router advertisement suppression 435
- Displaying VRRPv3 statistics 435
- Clearing VRRPv3 statistics 437
- VRRP-Ev3 Overview 438
- Enabling IPv6 VRRP-Ev3 438
- Configuring MD5 authentication on IPv6 VRRP-Ev3 interfaces 439
- VRRP-E load-balancing using short-path forwarding 440
- Displaying and clearing VRRP-Ev3 statistics 442

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.

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. To globally enable VRRPv3, enter the **ipv6 protocol vrrp** command.

   ```
   device(config)# ipv6 protocol vrrp
   ```

3. Enter the **interface ve** command with an associated VLAN number.

   ```
   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.

4. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2018)# ipv6 address 2001:2018:8192::125/64
   ```

5. 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.

6. 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.

7. 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)# 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 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. To globally enable VRRP, enter the **protocol vrrp** command.

   ```
   device(config)# protocol vrrp
   ```

3. Configure the Ethernet interface.

   ```
   device(config)# interface ethernet 1/6
   ```

   In this example, Ethernet interface configuration mode is entered and the interface is assigned with a slot/port number of 1/6.

4. Enter an IPv4 address for the interface using the **ip address** command.

   ```
   device(conf-if-eth-1/6)# ip address 192.168.5.2/24
   ```

5. Enter the **vrrp-group** command with a number to assign a virtual router group to the device and a version to configure VRRPv3.

   ```
   device(conf-if-eth-1/6)# vrrp-group 10 version 3
   ```

   In this example, a VRRPv3 group is assigned and VRRP group configuration mode is entered.

6. 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.

7. 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)# 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
```

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. To globally enable VRRPv3, enter the **ipv6 protocol vrrp** command.

   ```
   device(config)# ipv6 protocol vrrp
   ```

3. Enter the **interface ve** command with an associated VLAN number.

   ```
   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.

4. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2018)# ipv6 address 2001:2018:8192::125/64
   ```

5. 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.

6. 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.

7. 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 ethernet 1/5 priority 15
```

8. 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.

9. 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)# 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
```

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. Enable IPv6 VRRP-E.

   ```
   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.

3. Enter the **interface ve** command with an associated vlan number.

   ```
   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.

4. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
   ```

5. 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.

6. Enter the **description** command to enter text that describes the virtual router group.

   ```
   device(config-vrrp-group-18)# description Product Marketing group
   ```

7. 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.

8. 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)# 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
```

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.

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.

   ```
   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.

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. To enable VRRP globally enter the **protocol vrrp** command.

   ```
   device(config)# protocol vrrp
   ```

3. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config)# interface ve 2018
   ```

4. 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
   ```

5. 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)# protocol vrrp
device(config)# interface ve 2018
device(config-ve-2018)# vrrp-group 10 version 3
device(config-vrrp-group-10)# use-v2-checksum
```

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. Globally enable VRRP-Ev3.

```
device(config)# ipv6 protocol vrrp-extended
```

3. Enter the **interface ve** command with an associated VLAN number.

```
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.

4. 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)# ipv6 protocol vrrp-extended
device(config)# interface ve 2019
device(config-ve-2019)# ipv6 vrrp-suppress-interface-ra
```

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

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. 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

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.

Clearing VRRPv3 statistics

VRRPv3 session counters can be cleared using a CLI command.

Ensure that VRRPv3 is configured and enabled in your network.

1. Enter the **end** command, if required, to return to privileged EXEC mode.
2. 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. To globally enable VRRP-Ev3, enter the **ipv6 protocol vrrp-extended** command.

   ```
   device(config)# ipv6 protocol vrrp-extended
   ```

3. Enter the **interface ve** command with an associated virtual Ethernet (VE) interface number.

   ```
   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.

4. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-if-ve-2019)# ipv6 address 2001:2019:8192::122/64
   ```

5. 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.

6. 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.

7. 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)# 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.

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
```

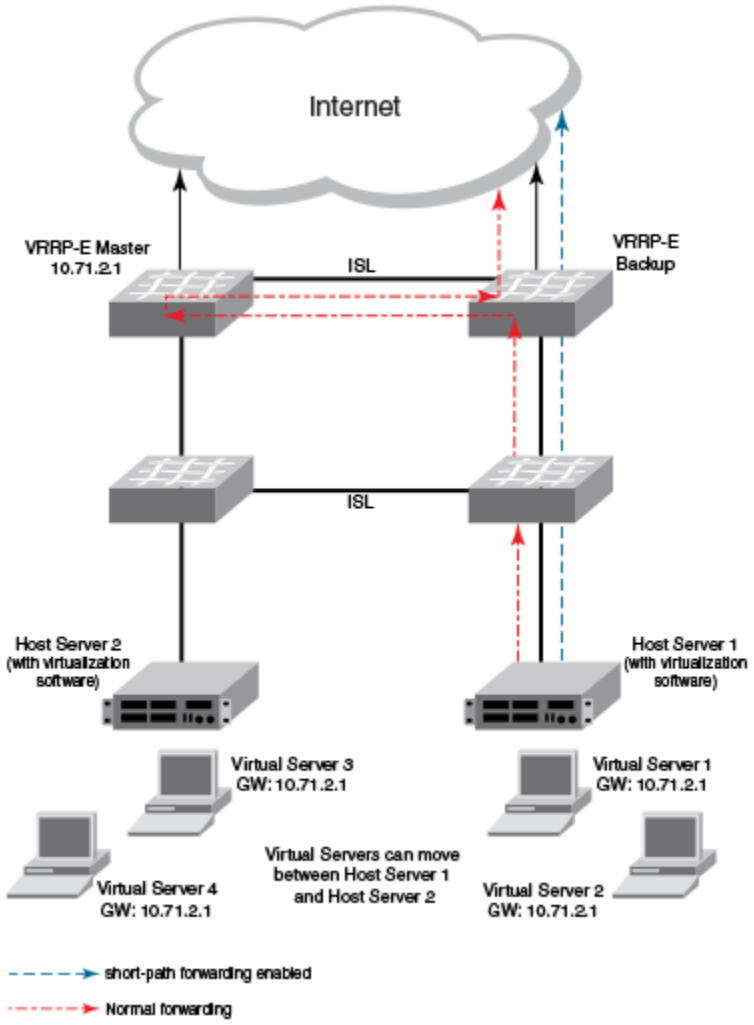
VRRP-E load-balancing using short-path forwarding

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.

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 59 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. Globally enable VRRP-Ev3.

   ```
   device(config)# ipv6 protocol vrrp-extended
   ```

3. Enter the **interface ve** command with an associated VLAN number.

   ```
   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.

4. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2019)# ipv6 address 2001:2019:8192::122/64
   ```

5. 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.

6. 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)# 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
```

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 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.

3. Enter the **show ipv6 vrrp 19 detail** command.

```
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.

4. 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
```

5. 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 Layer 3 Gateway

- Overview....445
- Configuring VXLAN Layer 3 gateway....452
- Example show and clear commands for VXLAN Layer 3 gateway....459
- BD VE support for VXLAN Layer 3 gateway....463

Overview

SLX routers support VXLAN Layer 2 gateway functionality. By acting as VXLAN Layer 3 gateways, SLX routers are capable of routing Layer 3 traffic while also terminating VXLAN tunnels.

To support Layer 3 functionality, a virtual Ethernet (VE) interface must be configured over either a VLAN or a bridge domain (BD) that contains both VXLAN tunnel members and attachment circuit (AC) end-point members. Such a VE (also known as "VE over VXLAN" or "VXLAN VE") can route and switch VXLAN traffic simultaneously.

With VXLAN Layer 3 gateways over VLANs/BDs, both static/EVPN and single/logical, the following options are supported:

- Single VTEP, static VLAN
- Single VTEP, static BD
- Single VTEP, EVPN VLAN
- Single VTEP, EVPN BD
- Logical VTEP, EVPN VLAN
- Logical VTEP, EVPN BD

The following table lists and describes the support for a variety of functionalities available under VXLAN Layer 3 gateway.

TABLE 41 VXLAN Layer 3 gateway support for functionalities

Functionality	Description	Comments
Routing protocols	Routing protocols cannot be enabled on a VE configured as a VXLAN Layer 3 gateway.	No routing protocols (such as OSPFor ISIS) are supported on such a VE.
VRF: VRF-lite/Multi-VRF)	A VE over VXLAN can be part of a nondefault VRF.	L3VPN-VRF is not yet supported under Logical VTEP.
ECMP	Support ECMP paths (8) for tunnel routing.	More than eight is not restricted but is not supported. The topology must be such that more than eight paths are not present for a VXLAN tunnel.
Statistics	Tunnel statistics are supported.	By default, statistics are enabled for both directions. If hardware resources are not available, then "N/A" is displayed.
BFD	BFD is not supported.	BFD is not supported for static tunnels.
VRRP	VRRPe source IP address/EVPN-MCT is not supported.	CLI configuration is not restricted.
MTU	MTU value is not configurable.	MTU is based on an IP interface MTU. If the packet is bigger than the IP interface MTU minus the VXLAN header, the packet is dropped.
TTL	TTL value is not configurable.	Default TTL value is 64.

TABLE 41 VXLAN Layer 3 gateway support for functionalities (continued)

Functionality	Description	Comments
QoS	QoS is not configurable.	Default QoS value is 0, which is applied to the DSCP field of the IP header.
-Tunnel uniform/pipe mode for TTL/QoS	Tunnel mode (uniform or pipe mode) is not configurable	By default, tunnel mode is pipe mode: in both directions, DSCP/TTL values are not carried from or to the native packet. In the VLAN-to-VXLAN direction, the default tunnel QoS/TTL values are used.
Exporting VE-over-VXLAN interface IP address using other protocols	Routing protocols running on other IP interfaces can export the VE-over-VXLAN IP address as connected routes.	A VE with VXLAN tunnels is treated as a directly connected subnet. This VE does not support protocols, as described above. However, as there is a connected subnet, reachability to this VE can be advertised through protocols such as OSPF, ISIS, and so on, configured as part of other Layer 3 interface configurations.
Ping/Traceroute	Ping and Traceroute are supported.	Ping supports traffic from/to VXLAN tunnels.
ARP	Dynamic ARP learning is supported in the VXLAN VE.	
Proxy ARP	Proxy ARP is not supported.	Proxy ARP configuration is not restricted, but the functionality is not supported.
Static ARP	Static ARP is supported.	Static ARP to an IP address reachable through a VXLAN tunnel is supported. The interface in the static ARP must be configured as the VE interface to which the host on the VXLAN tunnel is connected.
IPv6	IPv6 is supported.	
Static routes	Static routes are supported.	A static route can be configured to an IP address that is reachable through a VXLAN tunnel.
RPF	Reverse path forwarding (RPF) is not supported in the VXLAN VE.	RPF configuration is not restricted, but RPF functionality is not supported.
Multicast	Layer 3 multicast is not supported.	
PBR	Policy-based routing (PBR) is not supported.	ACL/PBR for native packets is not supported.
HA/ISSU	Hitless HA/ISSU is not supported.	Traffic hits are observed.
Inter-overlay routing	• IP routing is allowed from VE over VXLAN to VLAN-VE and vice versa. • IP routing is allowed from one VE-over-VXLAN tunnel to another. • IP routing is allowed between any other tunnel type (such as GRE/IP tunnel/MPLS-based pseudowire tunnels) to a VE-over-VXLAN tunnel. • Inter-VRF routing is not supported.	
Interoperability	The Layer 3 gateway interoperates with other SLX platforms in extension mode.	Interoperability with other VXLAN-supporting devices/hypervisors is not restricted but is not supported.
CAM profile	VXLAN L2/L3 gateway is supported only in the VxlanExtended TCAM profile	The configuration is not restricted in other profiles, but functionality is not supported.
Layer 2 gateway functionality	All the present Layer 2 gateway functionalities are supported on the Layer 3 gateway.	Only static VXLAN tunnels with regular VTEPs are supported; logical VTEP tunnels are not supported.

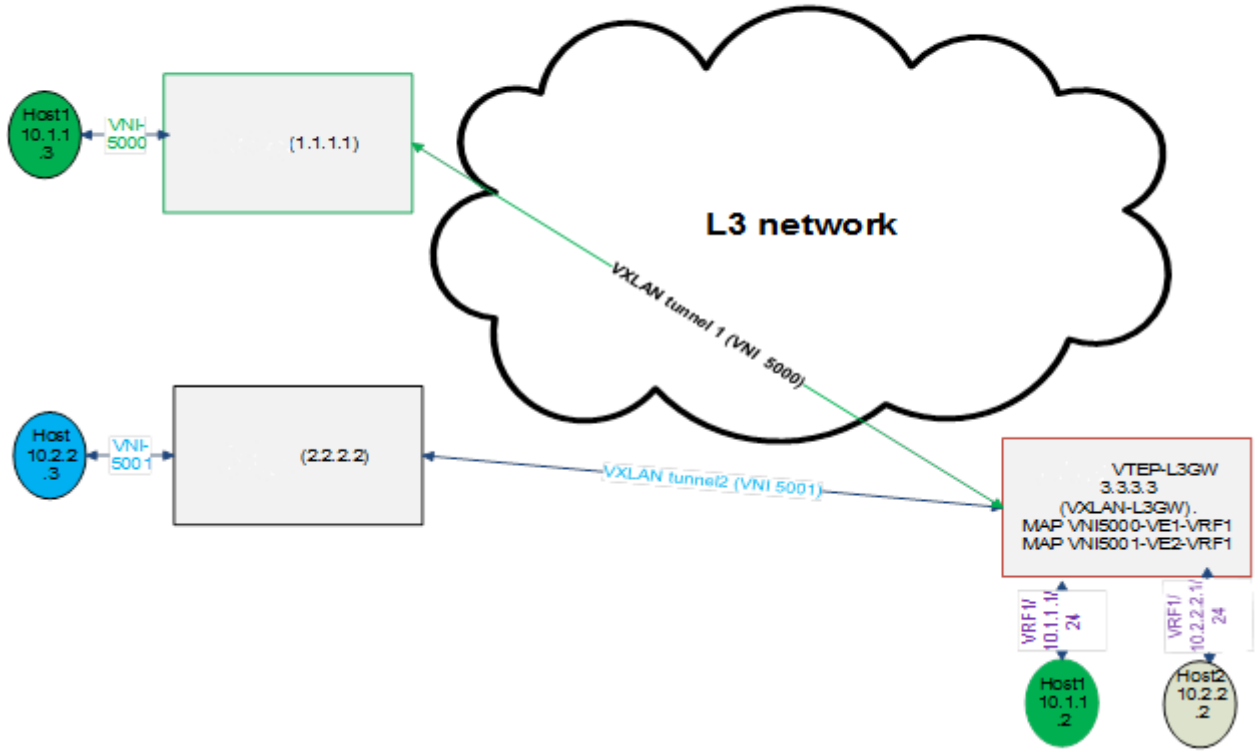
VXLAN Layer 3 gateway modes

Single-VTEP static VLAN/VXLAN-VE Layer 3 gateway

This is the most basic mode of VXLAN Layer 3 gateway (L3GW). In this mode of the L3GW, the VE is configured directly on the VLAN associated with the VXLAN Virtual Tunnel End Point (VTEP), which is mapped to the tunnel VXLAN Network Identifier (VNI).

One or more attachment circuit (AC) endpoints can be associated with the VLAN that is mapped to the VXLAN VE, or the VLAN can just contain the tunnel as its only member. The VXLAN tunnel is static, meaning that it is configured manually. The following figure illustrates a static L3GW topology.

FIGURE 60 Static L3GW topology



The scenario for this topology is as follows:

- A customer has two subnets, 10.1.1.0/24 and 10.2.2.0/24, which have hosts 10.1.1.3 (VNI 5000) and 10.2.2.3 (VNI 5001), respectively. The VRF of the customer is configured as VRF1 on the SLX nodes.
- Also on the SLX nodes, VLAN 10 (configured with routing VE1) maps to VNI 5000, and VLAN 20 (configured with routing VE2) maps to VNI 5001.
- Host 10.1.1.2 (Ethernet 1/1, VLAN10) maps to VNI 5000, and Host 10.2.2.2 (Ethernet 1/2, VLAN 20, maps to VNI 5001).

- If Host 10.1.1.3 must communicate with Host 10.2.2.2, packets must be routed at the VRF1 level. Packets come in on VXLAN tunnel 1 (VNI 5000) and are decapsulated and sent to interface Ethernet 1/1 upon ARP resolution, or an ARP resolution is attempted if the ARP is not resolved.

Single-VTEP static BD/VXLAN-VE

This scenario is the same as for single-VTEP static VLAN/VXLAN-VE L3GW, but in this case the VE is configured on a bridge domain (BD) that is extended over the VXLAN tunnel.

EVPN-based VXLAN Layer 3 gateway

EVPN VXLAN-L3GW functionality support is one of the several key features under the larger umbrella of IP Fabrics.

The following sections describe two features for Layer 3 routing.

IP-MAC routes on a single VTEP

Similar to the normal MAC routes that are exported and installed by EVPN BGP extensions, IP-MAC routes are also exported and installed on the remote nodes. The components of this scenario are detailed here.

BGP MAC/IP routes

This kind of route represents L3-to-L2 mapping, which is basically through ARP or ND. Static, dynamic ARP/ND entries are both exported to remote PEs and get installed as host routes. IPv4/IPv6 addresses that are configured on VE interfaces are also exported.

Upon ARP learning/gleaning/snooping on a local PE, ARP/ND information is exported to its EVPN BGP peers. The information mainly includes the following: MAC, IP/IPv6, L2-VNI, and L3-VNI. Such imported ARP/ND routes are installed or withdrawn as host routes in the hardware on the remote nodes. In the control plane they are available through the ARP suppression cache, which could be further used to reply for further ARP requests from hosts attached to the remote PE.

The packet path is as follows:

- When traffic that is bound to a remote host is received on the ingress PE GW, no ARP request is generated, as the route table already has the hardware host entry to forward or route the traffic to the destined host. This situation prevents ARP flooding and further processing.
- Packets get routed on the ingress PE itself, and then are switched all the way to the destination host, through the egress PE. Because routing occurs on the ingress PE and switching occurs on the remote PE, this type of forwarding is also termed "asymmetric routing."

On the nondefault VRF, the ARP/ND exports can have two subscenarios, depending on whether L2-VNI is extended on that PE or not:

- The imported IP-MAC route can be resolved against the L2-VNI if the L2-VNI is extended over the VXLAN tunnel, in which case the packet path is similar to that described previously.
- If the L2-VNI is not extended over the tunnel on that PE, the IP-MAC route is resolved against the L3-VNI. In this case the packet path followed is similar to a prefix routes path using L3-VNI, as described in the following section.

On the default VRF, formal host IP forwarding is done.

Layer 3 VNI on a single VTEP

For multitenant scenarios using VRFs in data centers, the L3-VNI identifies a particular tenant VRF across a VXLAN-EVPN tunnel. As the name suggests, L3-VNI is used mainly for routing purposes and in short identifies the tenant VRF.

BGP IP prefix routes on VRFs are exported to the remote PE by means of EVPN (Type-5). The information mainly includes the following: Egress-PE-GW-MAC, IP/IPv6 prefix route, and L3-VNI.

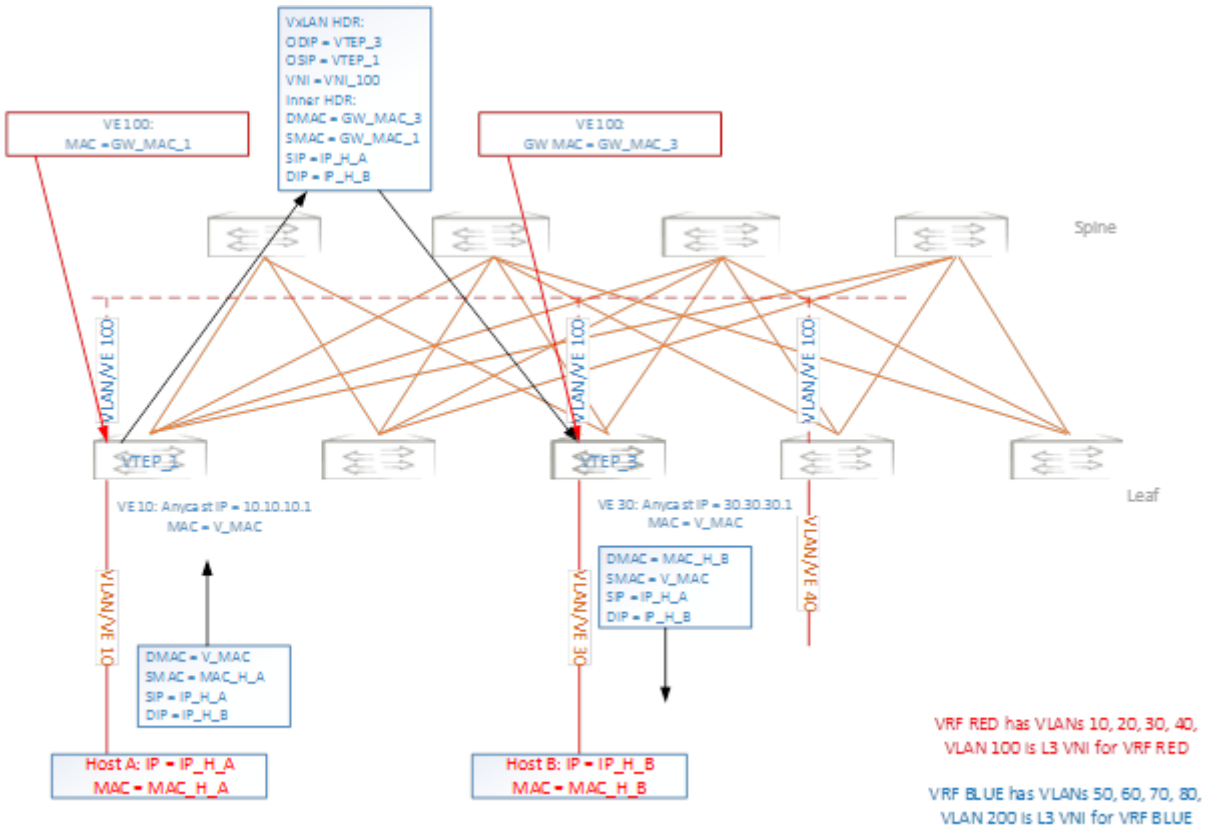
Such imported IP prefix routes are imported to VRFs and are installed as VRF routes, with the VXLAN tunnel having L3-VNI as the outgoing port and remote PE-GW-MAC as the destination MAC within the inner payload L2 header.

The packet path is as follows:

- The Layer 3 routing traffic that is originated on a particular VRF is terminated on the ingress PE gateway. As part of the L3 routing within the tenant VRF on the Ingress PE, the L3 packet is carried over the VXLAN tunnel to the egress PE by means of L3-VNI. The payload packet (L3) is always marked with the egress PE as the next hop.
- When the packet arrives at the egress PE, the outer header L3-VNI is used as an identifier to the tenant VRF, and the inner packet gets routed within this tenant VRF context.
- Because routing takes place on both the ingress and egress PE, this routing is also termed "symmetric routing."

The following figure illustrates this topology.

FIGURE 61 Layer 3 VNI on a single VTEP

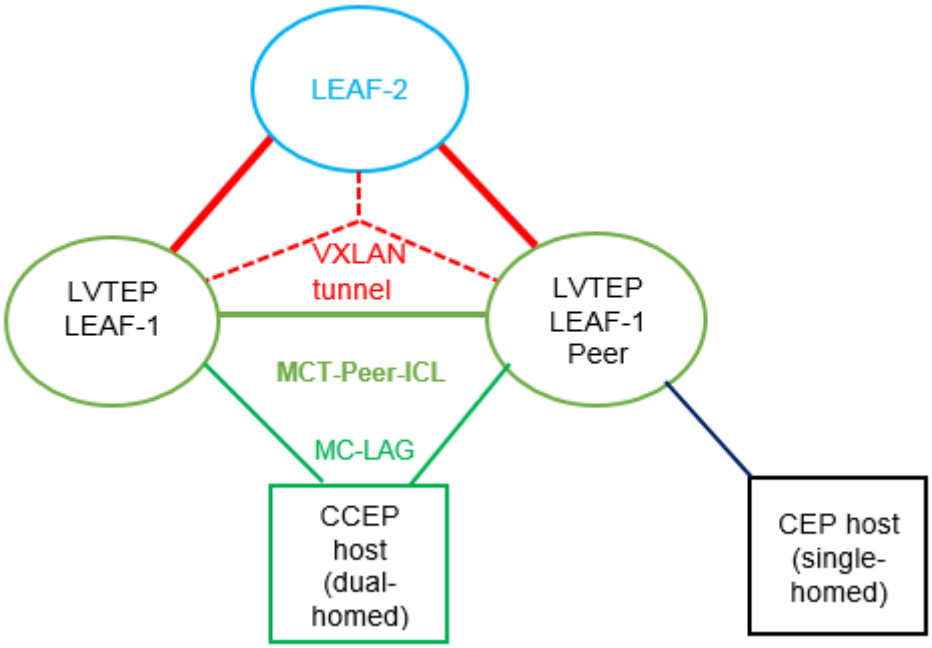


EVPN-based VXLAN Layer 3 gateway on LVTEP

This section discusses the Layer 3 functionality support on such a logical VTEP (LVTEP).

The following figure illustrates a VXLAN LVTEP topology.

FIGURE 62 VXLAN LVTEP topology



The LVTEP is formed through MCT peering (spoke-PW-peer) between Leaf-1 and its peer node, Leaf-12-peer, to provide redundancy for a VXLAN leaf node.

A VXLAN tunnel is created between such an LVTEP leaf and a remote leaf. The source IP address of the VXLAN tunnel is the same on both nodes. Therefore, the tunnel has a single tunnel representation on the remote leaf (Leaf-2 in the figure). The logical connection of the tunnel is shown as the dotted red line.

The single tunnel on Leaf-2 has two underlay paths to reach Leaf-1 and the Leaf-1 peer. Any traffic southbound from Leaf-2 is load balanced and can end up in either of the LVTEP peers.

IP-MAC routes on LVTEP

Similar to the single-VTEP case, the normal MAC-IP routes are exported and installed by EVPN BGP extensions on LVTEP between the leaf nodes, providing for the following behavior:

- Only one of the LVTEP peers that learns ARP (source LVTEP node) exports the route to the remote leaf. Such an imported route is installed as a host route that is pointed to the VXLAN tunnel, which has two underlay paths.
- The source LVTEP also syncs the ARP route to its LVTEP peer over ICL as part of MCT. These routes are installed pointing to the ICL interface (PW). Such synced IP-MAC routes are not readvertised to the VXLAN peer.
- The remote leaf exports its IP-MAC routes to both the LVTEP peers, and both peers install the routes in hardware—as host routes pointing to the local VXLAN tunnel toward the remote leaf.

A BGP MAC/IP route represents L3-to-L2 mapping, which is basically ARP or ND. Static and dynamic ARP/ND entries are exported to remote PEs and get installed as host routes. IPv4/IPv6 addresses that are configured on VE interfaces are also exported.

Upon ARP learning/gleaning/snooping on a local PE, ARP/ND information is exported to its EVPN BGP peers. The information mainly includes the following: MAC, IP/IPv6, L2-VNI, L3-VNI, and ESI segment. (In VXLAN, the ESI segment ID is always 0.)

Such imported ARP/ND routes are installed or withdrawn as host routes in the hardware on the remote nodes. In the control plane they are available through the ARP suppression cache, which can be further used to reply for further ARP requests from hosts that are attached to the remote PE.

The packet path is as follows:

- When traffic bound to a remote host is received on the Ingress PE GW, no ARP request is generated, as the route table already has the hardware host entry to forward/route the traffic to the destined host. This situation prevents ARP flooding and further processing.
- Packets get routed on the ingress PE itself, and then are switched all the way to the destination host, through the egress PE. Because routing occurs on the ingress PE and switching on the remote PE, this type of forwarding is also termed "asymmetric routing."

On the nondefault VRF, the ARP/ND exports can have two subscenarios, depending on whether L2-VNI is extended on that PE or not:

- The imported IP-MAC route could be resolved against L2-VNI if L2-VNI is extended over the VXLAN tunnel, in which case the packet path is similar to the one described previously.
- If L2-VNI is not extended over tunnel on that PE, the IP-MAC route is resolved against L3-VNI, in which case the packet path followed is similar to the prefix routes path using L3-VNI.

On the default VRF, normal host IP forwarding always occurs.

L3-VNI on LVTEP

Similar to the single-VTEP case, the IP prefix routes are also exported and installed by EVPN BGP extensions on LVTEP between the leaf nodes (with Type-5 routes), providing for the following:

- LVTEP peers export the IP prefix routes over BGP EVPN to remote leaf peer(s). Such imported routes are installed as prefix routes pointing to the VXLAN tunnel, which has two underlay paths.
- The IP prefix routes are also synced across the LVTEP peers on an MCT-ICL link and are installed as pointing to the ICL (PW). Such synced routes are not advertised to VXLAN peers.
- The remote leaf exports its IP prefix routes to both the LVTEP peers, both of which install the routes in hardware as network/prefix routes pointing to the local VXLAN tunnel.

BGP IP prefix routes on VRFs are exported to the remote PE over EVPN (Type-5). The information mainly includes the following: Egress-PE-GW-MAC, IP/IPv6 Prefix route, L3-VNI, ESI segment. (In VXLAN, the ESI segment ID is always 0.)

Such imported IP prefix routes are imported to VRFs and installed as VRF routes, with the VXLAN tunnel having L3-VNI as the outgoing port and remote PE-GW-MAC as the destination MAC with in the Inner Payload L2 header.

The packet path is as follows:

- The L3/routing traffic originated on a particular VRF is terminated on the ingress PE gateway. As part of the L3 routing with in the tenant VRF on the ingress PE, the L3 packet is carried over the VXLAN tunnel to the egress PE over L3-VNI. The payload packet (L3) is always marked with the egress PE as the next-hop.
- When the packet arrives at the egress PE, the outer header L3-VNI is used as an identifier to the tenant VRF, and the inner packet gets routed within this tenant VRF context.
- Because routing takes place on both the ingress and egress PE, this is also termed "symmetric routing."

Configuring VXLAN Layer 3 gateway

Configuring TCAM profiles to support Layer 3 gateway

Layer 3 gateway requires a hardware TCAM profile that supports VXLAN, as illustrated in this task.

ATTENTION
This profile must be configured for all the remaining tasks in this chapter.

1. Enter global configuration mode and enter the **hardware** command.

```
device# configure terminal
device(config)# hardware
```

2. In hardware configuration mode, enter the **profile tcam** command and specify **vxlan-ext**.

```
device(config-hardware)# profile tcam vxlan-ext
```

3. Save the running configuration to the startup configuration.

```
device# copy running-config startup config
```

4. Reboot the device.

Configuring a single-VTEP static VLAN/VE Layer 3 gateway

Follow these steps to configure a single-VTEP static VLAN/VE Layer 3 gateway.

1. Configure the TCAM profile to support L3GW. Complete the steps in Configuring TCAM profiles to support Layer 3 gateway on page 452.
2. Configure a loopback address.

```
interface Loopback 100
 no shutdown
 ip address 40.40.40.1/32
```

3. Configure an IP interface to be the interface of the VXLAN tunnel.

```
interface Ethernet 0/10
 ip proxy-arp
 ip address 50.50.50.1/24
 no shutdown
```

4. Create a VLAN.

```
vlan 500
```

5. Configure an attachment circuit (AC) endpoint.

```
interface Ethernet 0/20
 switchport
 switchport mode trunk
 switchport trunk allowed vlan add 500
 switchport trunk tag native-vlan
 no shutdown
```

6. Configure the VTEP.

```
overlay-gateway test
 type layer2-extension
 ip interface Loopback 100
 map vlan 500 vni 15000
 activate
 site VCS_2
 ip address 40.40.40.2  << This must be the remote-end loopback IP address and be reachable
 extend vlan add 500
```

7. Configure VE over VXLAN by configuring a VE over the VLAN associated with VXLAN.

```
vlan 500
 router-interface ve 500

int ve 500
 ip address 15.15.15.1/24
 ipv6 address 1001::1/64
 no shutdown
```

Configuring a single-VTEP static BD/VE Layer 3 gateway

Follow these steps to configure a single-VTEP static BD/VE Layer 3 gateway.

1. Configure the TCAM profile to support L3GW. Complete the steps in Configuring TCAM profiles to support Layer 3 gateway on page 452.
2. Configure a loopback address.

```
interface Loopback 100
 no shutdown
 ip address 40.40.40.1/32
```

3. Configure an IP interface to be the interface of the VXLAN tunnel.

```
interface Ethernet 0/10
 ip proxy-arp
 ip address 50.50.50.1/24
 no shutdown
```

4. Create a VLAN.

```
vlan 500
```

5. Configure an attachment circuit (AC) endpoint with a logical interface.

```
interface Ethernet 0/20
 switchport
 switchport mode trunk-no-default
 logical-interface eth 0/20.500 vlan 500
 no shut
```

6. Configure a bridge domain (BD).

```
bridge-domain 500 p2mp
 logical-interface eth 0/20.500
 pw-profile default
 bpdu-drop-enable
 local-switching
```

7. Configure the VTEP.

```
overlay-gateway test
 type layer2-extension
 ip interface Loopback 100
 map vlan 500 vni 15000
 activate
 site VCS_2
 ip address 40.40.40.2  << This must be the remote-end loopback IP address and be reachable
 extend bridge-domain add 500
```

8. Configure VE over VXLAN by configuring a VE over the VLAN that is associated with VXLAN.

```
vlan 500
 router-interface ve 500

int ve 500
 ip address 15.15.15.1/24
 ipv6 address 1001::1/64
 no shutdown
```

Configuring an EVPN Layer 3 gateway for MAC IP routes

Do the following to configure a Layer 3 gateway for MAC IP routes.

This configuration is similar to that for Layer 2 EVPN MAC routes.

1. Configure the TCAM profile to support L3GW. Complete the steps in Configuring TCAM profiles to support Layer 3 gateway on page 452.
2. Create VLANs to be extended on the tunnel.

```
vlan 1-4
```

3. Configure an EVPN instance.

```
evpn r1
 route-target both auto
 rd auto
 vlan add 2-4
```

4. Configure BGP.

```
router bgp
  local-as 100
    neighbor 98.0.0.1 remote-as 100
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  !
  address-family l2vpn evpn
    graceful-restart
    neighbor 98.0.0.1 encapsulation vxlan
    neighbor 98.0.0.1 activate
```

5. Configure a loopback interface.

```
interface Loopback 1
 no shutdown
 ip ospf area 0
 ip address 1.2.3.4/32
```

6. Configure a tunnel interface.

```
interface Ethernet 0/4
 ip ospf area 0
 ip proxy-arp
 ip address 98.0.0.2/24
 no shutdown
```

7. Configure the overlay gateway.

```
overlay-gateway g1
type layer2-extension
ip interface Loopback 1
  map vni auto
activate
```

Configuring an EVPN Layer 3 VNI

Follow these steps to configure an EVPN Layer 3 VNI.

Layer 3 VNI is used to support VRFs.

1. Configure the TCAM profile to support L3GW. Complete the steps in Configuring TCAM profiles to support Layer 3 gateway on page 452.
2. Create VLANs to be extended on the tunnel.

```
vlan 1-4
```

3. Configure an EVPN instance.

```
evpn r1
 route-target both auto
 rd auto
 vlan add 2-4
```

4. Configure BGP.

```
router bgp
  local-as 100
    neighbor 98.0.0.1 remote-as 100
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  !
  address-family l2vpn evpn
    graceful-restart
    neighbor 98.0.0.1 encapsulation vxlan
    neighbor 98.0.0.1 activate
```

5. Configure a loopback interface.

```
interface Loopback 1
 no shutdown
 ip ospf area 0
 ip address 1.2.3.4/32
```

6. Configure a tunnel interface.

```
interface Ethernet 0/4
 ip ospf area 0
 ip proxy-arp
 ip address 98.0.0.2/24
 no shutdown
```

7. Configure the overlay gateway.

```
overlay-gateway g1
type layer2-extension
ip interface Loopback 1
  map vni auto
activate
```

8. Configure a VRF.

```
vrf red
rd 5:50
evpn irb ve 100 <peer-gateway> <--Identifies the L3-VNI
address-family ipv4 unicast
  route-target export 5:100 evpn
  route-target import 5:100 evpn
!
address-family ipv6 unicast
  route-target export 5:100 evpn
  route-target import 5:100 evpn
```

9. Configure the L3-VNI Integrated Routing and Bridging (IRB) instance. (An IP address is not required.)

```
vlan 100  <-- The L3-VNI is a VLAN as a result of auto map mode
router-interface Ve 100
```

10. Alternatively, configure the L3-VNI IRB by using a BD. (The L3-VNI is 4K+BD-ID as a result of auto mapping.)

```
bridge-domain 500 p2mp
 router-interface ve 100
!
interface Ve 100
vrf forwarding red
no shutdown
```

Configuring an EVPN LVTEP for MAC IP routes

Follow these steps to configure an EVPN LVTEP for MAC IP routes.

This configuration is similar to that for Layer 2 EVPN MAC routes.

1. Configure the TCAM profile to support L3GW. Complete the steps in Configuring TCAM profiles to support Layer 3 gateway on page 452.
2. Create VLANs to be extended on the tunnel.

```
vlan 1-4
```

3. Configure an EVPN instance.

```
evpn r1
 route-target both auto
 rd auto
 vlan add 2-4
```

4. Configure BGP.

```
router bgp
 local-as 100
  neighbor 3.3.3.3 remote-as 101 <-- The VXLAN peer
  neighbor 40.40.100.50 remote-as 102 <-- The MCT peer
 address-family ipv4 unicast
 !
 address-family ipv6 unicast
 !
 address-family l2vpn evpn
  graceful-restart
  neighbor 3.3.3.3 encapsulation vxlan <-- The VXLAN peer
  neighbor 3.3.3.3 activate
  neighbor 40.40.100.50 encapsulation mpls <-- The MCT peer
  neighbor 40.40.100.50 activate
```

5. Configure a loopback interface for BGP neighborship.

```
interface Loopback 1
 no shutdown
 ip ospf area 0
 ip address 40.40.100.40/32
```

6. Configure a loopback interface for VXLAN. (MCT peers must have the same address.)

```
interface Loopback 2
 no shutdown
 ip ospf area 0
 ip address 2.2.2.2/32
```

7. Configure a tunnel interface.

```
interface Ethernet 0/4
 ip ospf area 0
 ip proxy-arp
 ip address 98.0.0.2/24
 no shutdown
```

8. Configure the overlay gateway.

```
overlay-gateway g1
 type layer2-extension
 ip interface Loopback 1
  map vni auto
 activate
```

9. Configure the MCT cluster.

```
cluster c1 1
 peer-interface Ve 45
 peer 40.40.100.50
 deploy
```

Configuring an EVPN Layer 3 VNI for LVTEP

This example configures an L3-VNI to support a VRF.

1. Configure the TCAM profile to support L3GW. Complete the steps in Configuring TCAM profiles to support Layer 3 gateway on page 452.
2. Instantiate VLANs to be extended on the tunnel.

```
vlan 1-4
```

3. Configure an EVPN instance.

```
evpn r1
 route-target both auto
 rd auto
 vlan add 2-4
```

4. Configure BGP.

```
router bgp
 local-as 100
  neighbor 3.3.3.3 remote-as 101 <-- The VXLAN peer
  neighbor 40.40.100.50 remote-as 102 <-- The MCT peer
 address-family ipv4 unicast
 !
 address-family ipv6 unicast
 !
 address-family l2vpn evpn
  graceful-restart
  neighbor 3.3.3.3 encapsulation vxlan <-- The VXLAN peer
  neighbor 3.3.3.3 activate
  neighbor 40.40.100.50 encapsulation mpls <-- The MCT peer
  neighbor 40.40.100.50 activate
!
```

5. Configure a loopback interface for BGP neighborship.

```
interface Loopback 1
 no shutdown
 ip ospf area 0
 ip address 40.40.100.40/32
```

6. Configure a loopback interface for VXLAN. (MCT peers must have the same address.)

```
interface Loopback 2
 no shutdown
 ip ospf area 0
 ip address 2.2.2.2/32
```

7. Configure a tunnel interface.

```
interface Ethernet 0/4
 ip ospf area 0
 ip proxy-arp
 ip address 98.0.0.2/24
 no shutdown
```

8. Configure the overlay gateway.

```
overlay-gateway g1
 type layer2-extension
 ip interface Loopback 1
  map vni auto
 activate
```

9. Configure the MCT cluster.

```
cluster c1 1
 peer-interface Ve 45
 peer 40.40.100.50
 deploy
```

10. Configure the VRF.

```
vrf red
rd 5:50
evpn irb ve 100 <peer-gateway> <-- Identifies the L3-VNI
address-family ipv4 unicast
  route-target export 5:100 evpn
  route-target import 5:101 evpn  <-- From the MCT peer
  route-target import 5:102 evpn  <-- From the VXLAN peer
!
address-family ipv6 unicast
```

11. Configure the L3-VNI IRB. (An IP address is not needed.)

```
vlan 100
 router-interface Ve 100
!
interface Ve 100
 vrf forwarding red
 no shutdown
```

Example show and clear commands for VXLAN Layer 3 gateway

This section presents example output of the **show** and **clear** commands that are useful in managing VXLAN Layer 3 gateway.

show overlay-gateway

The following is example output from the **show overlay-gateway** command.

```
device# show overlay-gateway
Overlay Gateway "VXLAN", ID 1, rbridge-ids 1
Admin state up
IP address 33.32.31.13 (loopback 1), Vrf default-vrf
Number of tunnels 6
Packet count: RX 17909 TX 1247
Byte count : RX (500125) TX 356626
```

show tunnel brief

The following is example output from the **show tunnel brief** command.

```
device# show tunnel brief
Tunnel 1, mode VXLAN, rbridge-ids 1
Admin state up, Oper state up
Source IP 33.32.31.13, Vrf default-vrf
Destination IP 33.32.31.10
Tunnel 2, mode VXLAN, rbridge-ids 1
Admin state up, Oper state up
Source IP 33.32.31.13, Vrf default-vrf
Destination IP 33.32.31.1
```

show tunnel

The following is example output from the **show tunnel** *ID* command.

```
device# show tunnel 1
Tunnel 1, mode VXLAN, rbridge-ids 1
Ifindex 2080374798, Admin state up, Oper state up
Overlay gateway "VXLAN", ID 1
Source IP 33.32.31.13 (loopback 1), Vrf default-vrf
Destination IP 33.32.31.10
Active next hop on rbridge1:
IP: 33.32.31.10, Vrf: default-vrf
Egress L3 port: Ve 10, Outer SMAC: 0027.f886.bb36
Outer DMAC: e41f.1343.97d2
Egress L2 Port: Po 11, Outer ctag: 10
BUM forwarder: yes
Packet count: RX 18492 TX 1242
Byte count : RX (NA) TX 356307
```

show vlan brief

The following is example output from the **show vlan brief** command, to verify VNI mapping.

```
device# show vlan brief
Total Number of VLANs configured : 1
Total Number of VLANs provisioned : 1
Total Number of VLANs unprovisioned : 0
VLAN Name State Ports Classification
(F)-FCoE (u)-Untagged, (t)-Tagged
(R)-RSPAN (c)-Converged
(T)-TRANSPARENT
30 VLAN0030 ACTIVE Po 11(t)
Tu 1(t) vni 100
Tu 2(t) vni 100
```

show mac-address-table

The following is example output from the **show mac-address-table** command with the **bridge-domain** keyword, to verify gateway MAC addresses.

```
device# show mac-address-table bridge-domain
VlanId/BD-Id   Mac-address           Type     State     Ports/LIF/peer-ip
629(B)            0011.2222.5555     Dynamic  Active    eth 1/3.100
629(B)            0011.2222.6666     Dynamic  Inactive  eth 1/1.500
629(B)            0011.2222.1122     Dynamic  Active    10.12.12.12
629(B)            0011.2222.3333     static   Inactive  po 5.700
629(B)            0011.0101.5555     Dynamic  Active    eth 1/2.400

Total MAC addresses : 5
```

show ip arp suppression-cache

The following is example output from the **show ip arp suppression-cache** command, to verify remote ARPs and NDs..

```
device# show ip arp suppression-cache
Flags: L  - Locally Learnt Adjacency
       R  - Remote Learnt Adjacency
       RS - Remote Static Adjacency
Vlan/Bd    IP              Mac             Interface                       Age           Flags
-------------------------------------------------------------------------------------------------
0100 (V)   10.10.10.11     0000.0a0a.0a0b X/X                          00:01:35        L
0101 (V)   10.10.10.98     0000.1111.0000 X/X                          Never           RS
0101 (V)   10.10.10.99     609c.9f5a.4d15 X/X                          Never           RS
0102 (V)   12.12.122.1     609c.9f5a.4715 X/X                          Never           RS
```

show bgp evpn

All the options under the **show bgp evpn** command are helptul.

The following is example output from the **show bgp evpn l3vni** command, for a specific VRF .

```
device# show bgp evpn l3vni vrf red

-------------------------------------------------------------------------
   L3VNI Prefix Origination Conditions for vrf (red)
-------------------------------------------------------------------------
Address Family under BGP : True
RD Configured            : True
IRB I/F Configured        : True (0x48000064)
IRB I/F Status            : False
IRB EVID Conigured        : True (100)
Router mac Exists        : True
Source VTEP              : 40.40.40.1
VTEP Active              : Active
IPv4 L3VNI Active             : Active
IPv6 L3VNI Active             : Inactive


-------------------------------------------------------------------------
   L3VNI Prefix Import Conditions for vrf (red)
-------------------------------------------------------------------------
Address Family under BGP : True
IRB I/F Configured       : True (0x48000064)
IRB EVID Configured      : True (100)
Router mac Exists        : True
IPv4 L3VNI Active             : Active
IPv6 L3VNI Active             : Inactive
```

The following is example output from the **show bgp evpn routes** command, with ARP specified.

```
device# show bgp evpn routes type arp

Total number of BGP EVPN ARP 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
Route Distinguisher: 40.40.100.50:32869
1      ARP:[0][0000.0a0a.0a0b]:[IPv4:10.10.10.11]
                          0.0.0.0         0          100        0      BL
         AS_PATH:
            L2 Label: 101 L3 Label: 100
         ESI : 00.000000000000000000
2      ARP:[0][609c.9f5a.4715]:[IPv4:15.143.15.1]
                          0.0.0.0         0          100        0      BL
         AS_PATH:
            L2 Label: 101 L3 Label: 100
         ESI : 00.000000000000000000
Route Distinguisher: 40.40.100.50:33769
3      ARP:[0][609c.9f5a.4715]:[IPv4:14.13.15.1]
                          0.0.0.0         0          100        0      BL
         AS_PATH:
            L2 Label: 1001 L3 Label: 100
         ESI : 00.000000000000000000
Route Distinguisher: 40.40.100.60:32869
4      ARP:[0][609c.9f5a.8d15]:[IPv4:6.6.2.5]
                          40.40.40.2      0          100        0      BE
         AS_PATH: 1000
            L2 Label: 101 L3 Label: 0
         ESI : 00.000000000000000000
```

The following is example output from the **show bgp evpn routes** command, with IPv4 prefixes specified.

```
device# show bgp evpn routes type ipv4-prefix briief
Total number of BGP EVPN Ipv4Prefix 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
Route Distinguisher: 5:50
*>  IP4Prefix:[0][14.13.15.0/24]
                       0.0.0.0         0          100        0      ?
*>  IP4Prefix:[0][15.143.15.0/24]
                       0.0.0.0         0          100        0      ?
*>  IP4Prefix:[0][16.16.16.0/24]
                       0.0.0.0         0          100        0      ?
Route Distinguisher: 5:100
*>  IP4Prefix:[0][17.17.17.0/24]
                       40.40.40.2      0          100        0      1000 ?
```

show tunnel statistics

The following is example output from the **show tunnel statistics** command.

```
device# show tunnel statistics
Tnl ID RXpackets TX packets RX bytes TX bytes
======== ============== ============== ============== ================
1      18573       1242       356307     356307
```

clear counters all

Use the **clear counters all** command to clear statistics on a gateway, including tunnel statistics.

clear overlay-gateway

Use the **clear overlay-gateway** command to clear statistics on an overlay gateway, including tunnel statistics.

BD VE support for VXLAN Layer 3 gateway

BD VE overview

An SLX device can act as a VXLAN Layer 3 gateway (L3GW) to connect different network segments. With L3GW, the device routes VXLAN Layer 2 and Layer 3 traffic over a VXLAN tunnel and conversely.

For Layer 3 forwarding, a virtual Etrhernet (VE) IP interface is configured under a bridge domain (BD). This VE interface can be part of a default or nondefault VRF. No routing protocols (such as OSPFor IS-IS) are supported over such a VE, which is a regular connected subnet. However, this subnet can be announced by routing protocols when it is configured on other IP interfaces in the device. L3GW is supported between SLX 9850 series platforms, or between those platforms and VDX 6740 series platforms that act as L3GW routers.

> NOTE
> While L3GW can be configured between SLX 9850 series platforms and NSX Hypervisor virtual machines (VMs), this configuration is not supported.

Feature components

The following components support this feature.

BD

A bridge domain is a Layer 2 broadcast domain that provides for the forwarding of VXLAN data packets. VXLAN Network Identifiers (VNIs) that identify VXLAN networks (VNs) must be mapped to BDs in 1:1 mode, so that a BD can function as a VXLAN network entity to transmit VXLAN traffic.

BD interface

A BD interface (BDIF) is a Layer 3 logical interface that is created for a BD. Configuring IP addresses for BDIF interfaces allows communication between VXLANs on different network segments and between VXLANs and non-VXLANs, and implements Layer 2 network access to a Layer 3 network.

VNI

A VNI is a VXLAN segment identifier similar to a VLAN ID. VMs on different VXLAN segments cannot communicate directly at Layer 2. A VNI identifies only one tenant. Even if multiple terminal users belong to the same VNI, they are considered as a single tenant. A VNI consists of 24 bits and supports a maximum of 16 M tenants. In distributed VXLAN gateway scenarios, a VNI can be a Layer 2 or a Layer 3 VNI.

A Layer 2 VNI is mapped to a BD in 1:1 mode for the intrasegment transmission of VXLAN packets. A Layer 3 VNI is bound to a VPN.

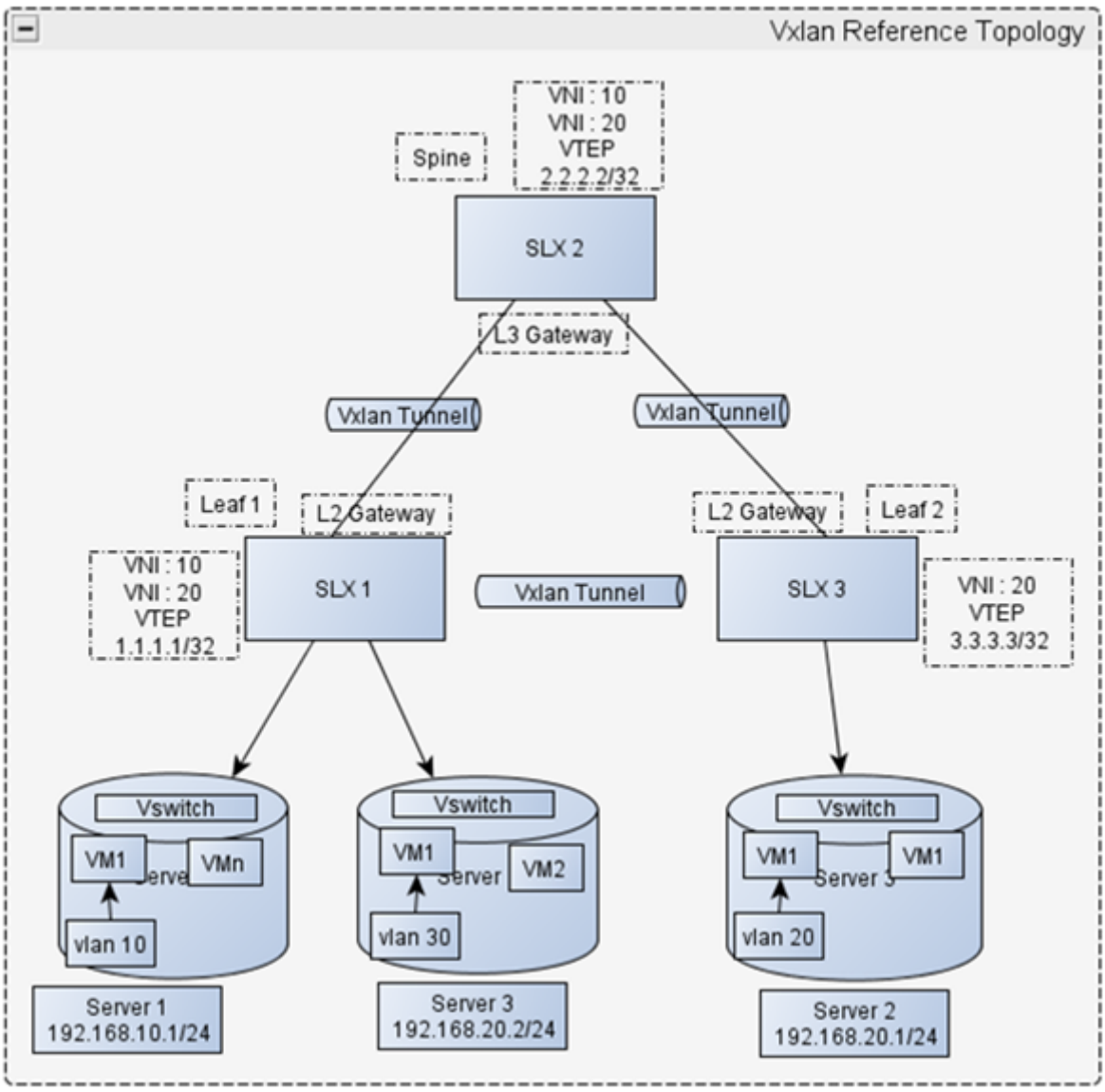
A VXLAN tunnel is identified by a pair of VXLAN tunnel endpoint (VTEP) IP addresses. A VXLAN tunnel is statically created after the user configures local and remote VNIs and VTEP IP addresses, and the tunnel goes up when the pair of VTEPs are reachable at Layer 3.

Topologies

The following topologies illustrate L3GW centralized and distributed scenarios.

The following figure illustrates a centralized VXLAN gateway scenario.

FIGURE 63 Centralized VXLAN gateway scenario

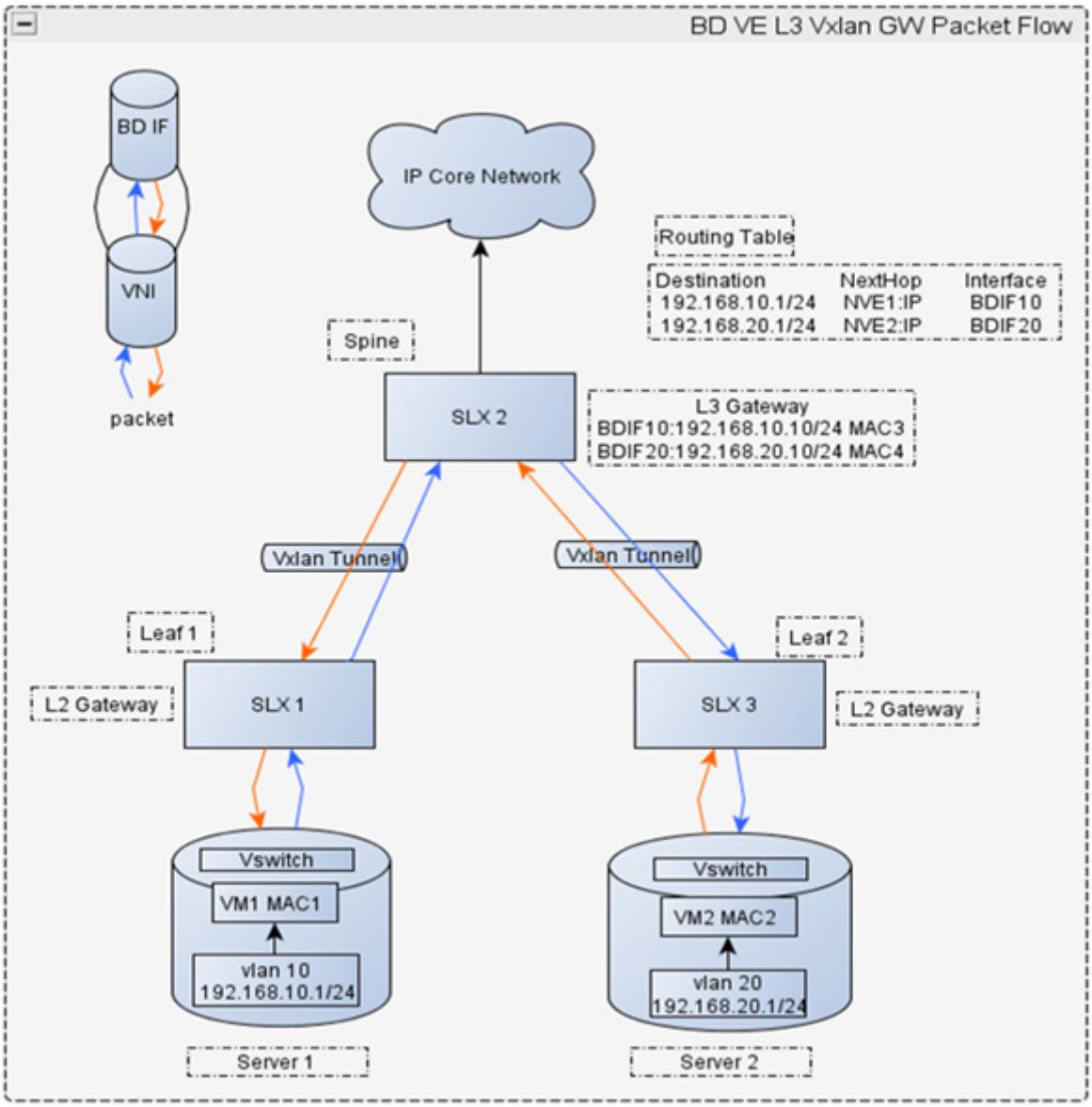


In the network shown above, Server 1 and Server 3 are deployed for SLX 1, and Server 2 is deployed for SLX 3. Server 1 and Server 2 reside on different network segments, whereas Server 2 and Server 3 reside on the same network segment. To allow VMs on Server 2 and Server 3 to communicate, VNIs and VTEP IP addresses must be configured to establish a VXLAN tunnel between SLX 1 and SLX 3. To allow VMs on Server 1 and Server 2 to communicate, VNIs and VTEP IP addresses must be configured to establish a VXLAN tunnel between SLX 1 and SLX 2 as well as between SLX 2 and SLX 3.

The IP interface connected to Server 1 on SLX 1 and the IP interface connected to Server 2 can be configured on a user-defined VRF.

The following figure illustrates a scenario for distributed VXLAN gateway packet forwarding.

FIGURE 64 Distributed VXLAN gateway packet forwarding



In the network shown above, the intersegment packet forwarding process is as follows:

1. After SLX 2 receives VXLAN packets, it decapsulates them and checks whether the destination MAC address in the inner packets is the MAC address of the Layer 3 gateway interface, BDIF10. If the destination MAC address is a local MAC address, SLX 2 forwards the packets to the Layer 3 gateway on the destination network segment and proceeds to Step 2. If the destination MAC address is not a local MAC address, Device 3 searches for the outbound interface and encapsulation information in the Layer 2 BD.
2. SLX 2 removes the Ethernet headers of the inner packets and parses the destination IP address. SLX 2 searches the routing table for the next-hop IP address (based on the destination) and the ARP entries (based on the next-hop IP address). SLX 2 uses the ARP entries to identify the destination MAC address, the outbound interface of the VXLAN tunnel, and the VNI. If the

outbound interface of the VXLAN tunnel and the VNI cannot be found, SLX 2 performs Layer 3 forwarding. If the VXLAN tunnel's outbound interface and VNI can be found, SLX 2 proceeds to Step 3.

3. SLX 2 encapsulates the VXLAN packets again, with the source MAC address in the Ethernet header of the inner packets as the MAC address of the Layer 3 gateway interface, BDIF20.

Layer 3 support: limitations and considerations

This section lists the limitations and considerations for this feature.

The IP next-hop for the underlay tunnel must not be through the subnet that is configured for the VE interface over the tunnel. However, this is not restricted in software.

The following table lists supported and unsupported features for VXLAN underlay.

TABLE 42 Supported and unsupported features for VXLAN underlay

Supported	Not supported
Both nondefault and default VRF configurations	Proxy ARP
ARP learning on the tunnel	L3 protocols (not enforced)
Static routes	RPF (not enforced)
Connected subnets, which can be redistributed over other protocols	VRRP/e (not enforced)
Pinging of connected subnet hosts over a VXLAN tunnel	L2 or L3 multicast (not enforced)
Traceroute	PDUs that exceed MTU size (if the packet to be sent over the VXLAN tunnel exceeds MTU size, the packet is dropped)
	Keepalive of any kind
	High availability or ISSU

The following table lists supported and unsupported features for routing.

TABLE 43 Supported and unsupported features for routing

Supported	Not supported
IP routing from a VE-over-VXLAN tunnel to a pure VE and conversely	IP routing between any other tunnel type (such as GRE/IP tunnel, MPLS-based pseudowire tunnels) to a VXLAN tunnel (not enforced; such configurations must be avoided)
IP routing from one VE-over-VXLAN tunnel to another	Inter-VRF

BD ID range

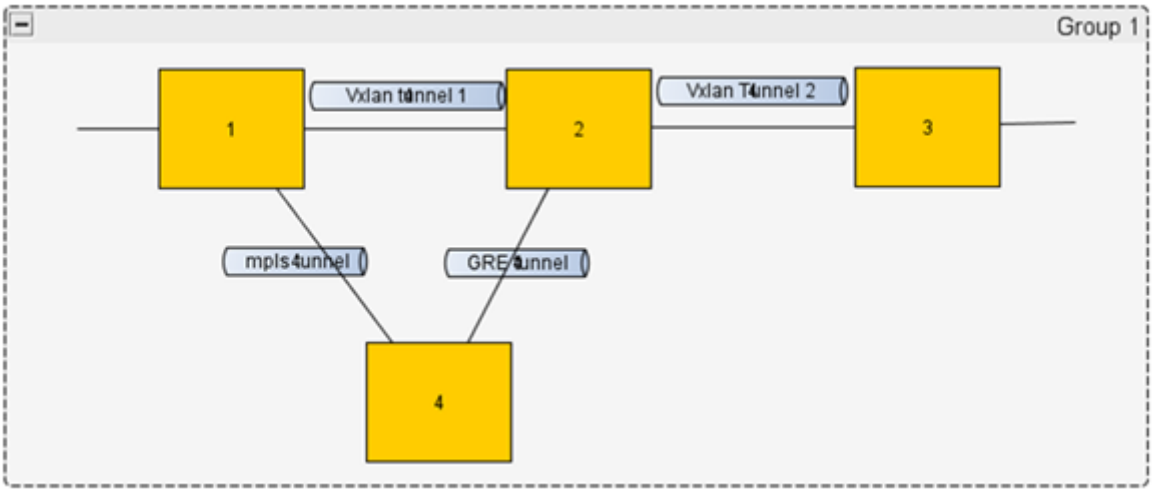
User-configured BD ID values range from 1 through 4 K.

Advanced routing scenarios

The following figure illustrates a topology that supports routing where the following scenarios are allowed:

- Routing between one VXLAN tunnel and another
- Routing between a VXLAN tunnel and an MPLS tunnel and vice-versa
- Routing between any other tunnel type (such as GRE/IP) and a VXLAN tunnel and vice-versa

FIGURE 65 Advanced routing scenarios



Configuring BD VE support for VXLAN Layer 3 gateway

Configuring a static VXLAN tunnel

Follow these steps to configure a static VXLAN tunnel.

1. Configure the TCAM profile to support L3GW. Complete the steps in Configuring TCAM profiles to support Layer 3 gateway on page 452.
2. Enter global configuration mode.

```
device# configure terminal
```

3. Enter the **overlay-gateway** command and specify the name of a gateway.

```
device(config)# overlay-gateway l3vxlangw
```

4. Enter the **type** command and confirm that Layer 2 extension is specified.

```
device(config-overlay-gw-l3vxlangw)# type l2-extension
```

   Layer 2 extension is the only type supported.

5. Enter the **ip interface loopback** command and specify a loopback interface.

```
device(config-overlay-gw-l3vxlangw)# ip interface loopback 1
```

   This specifies the loopback port number for the overlay gateway instance.

6. Enter the **map bridge-domain** command and map a bridge domain to a VNI.

```
device(config-overlay-gw-l3vxlangw)# map bridge-domain 100 to vni 10000
```

7. Enter the **site** command and specify a site.

```
device(config-overlay-gw-l3vxlangw)# site vtep1
```

8. Enter the **ip address** command and specify an IPv4 address for the site.

```
device(config-overlay-gw-l3vxlangw-site-vtep1)# ip address 1.1.1.1
```

9. Enter the **extend bridge-domain** command to add the VLAN, and exit to overlay-gateway configuration mode.

```
device(config-overlay-gw-l3vxlangw-site-vtep1)# extend bridge-domain add 100
device(config-overlay-gw-l3vxlangw)# exit
```

10. Enter the **activate** command to enable the VXLAN gateway.

```
device(config-overlay-gw-l3vxlangw)# activate
```

Example static VXLAN tunnel configurations

This section presents static VXLAN tunnel configurations on two peer SLX nodes.

Node A configurations

```
vrf red
rd 1:1
address-family ipv4 unicast
  ip route 92.92.92.0/24 11.0.1.1

bridge-domain 100 p2mp   <--Mapped BD router-interface Ve 100
pw-profile default
 bpdu-drop-enable
 local-switching

overlay-gateway l3vxlangw   <--Gateway configuration
type layer2-extension
ip interface Loopback 1
map bridge-domain 100 vni 10000
activate
site vtep1
ip address 1.1.1.1
extend bridge-domain add 100

interface Ve 100   <--Mapped BD interface
 vrf forwarding red
 ip proxy-arp
 ip address 11.0.1.2/24
 shutdown

interface Ethernet 0/5   <--Access interface
 vrf forwarding red
 ip proxy-arp
 ip address 91.91.91.1/24
 no shutdown

interface Ethernet 0/3   <--VXLAN tunnel interface
 ip proxy-arp
 ip address 10.0.1.2/24
 no shutdown

interface Loopback 1
 ip address 2.2.2.2/32
 no shutdown

ip route 92.92.92.2/32 11.0.1.1   <--Default VRF static route
ip route 10.10.10.0/24 next-hop-vrf red 9.9.9.2   <--Nondefault VRF static route
```

Node B configurations

```
vrf red
rd 1:1
address-family ipv4 unicast
  ip route 92.92.92.0/24 11.0.1.1

bridge-domain 100 p2mp   <--Mapped BD
 router-interface Ve 100
pw-profile default
 bpdu-drop-enable
 local-switching

overlay-gateway l3vxlangw   <--Gateway configuration
type layer2-extension
ip interface Loopback 1
map bridge-domain 100 vni 10000
activate
site vtep1
ip address 2.2.2.2
extend bridge-domain add 100

interface Ve 100   <--Mapped BD interface
 vrf forwarding red
 ip proxy-arp
 ip address 11.0.1.1/24
 shutdown

interface Ethernet 0/4   <--Access interface
 vrf forwarding red
 ip proxy-arp
 ip address 92.92.92.1/24
 no shutdown

interface Ethernet 0/3   <--VXLAN tunnel interface
 ip proxy-arp
 ip address 10.0.1.1/24
 no shutdown

interface Loopback 1
 ip address 1.1.1.1/32
 no shutdown

ip route 92.92.92.1/32 11.0.1.2   <--Static route
```

NOTE

Use the **map vni auto** command to map a BD ID to a VNI automatically.

Example BGP EVPN-based configurations

BGP EVPN configuration is required for automatic VXLAN tunnel discovery and the automatic mapping of VLANs/BDs to VNIs.

The following examples illustrate the configuration of peering interfaces, BGP, an overlay gateway, and EVPN.

Peering interfaces

The following example shows how to create a VLAN/BD and a VE interface and add the VLAN to the physical port connecting R1 and R2.

R1

```
vlan 100
 router-interface Ve 100
!

interface Ve 100
```

```
 ip ospf area 0
 ip proxy-arp
 ip address 10.0.0.1/24
 no shutdown
!

interface Ethernet 0/1   <--Change this according to the physical connection
 switchport
 switchport mode trunk
 switchport trunk allowed vlan all
 no shutdown
!
```

R2

```
vlan 100
 router-interface Ve 100
!
interface Ve 100
 ip ospf area 0
 ip proxy-arp
 ip address 10.0.0.2/24
 no shutdown
```

BGP

The following example shows how to configure a local AS and a remote neighbor IP address and activate the neighbor under EVPN address-family.

R1

```
router bgp
 local-as 100
 neighbor 10.0.0.2 remote-as 100
 address-family ipv4 unicast
 !
 address-family ipv6 unicast
 !
 address-family evpn
  neighbor 10.0.0.2 activate
 !
!
```

R2

```
router bgp
 local-as 100
 neighbor 10.0.0.1 remote-as 100
 address-family ipv4 unicast
 !
 address-family evpn
  neighbor 10.0.0.1 activate
 !
!
```

Overlay gateway

The following example shows how to configure a source VTEP loopback interface and an overlay gateway.

R1

```
interface Loopback 1
 ip address 1.2.3.4/32
 no shutdown
!

overlay-gateway g1
```

```
 type layer2-extension
 ip interface Loopback 1
 map vni auto
 activate
!
```

R2

```
interface Loopback 1
 ip address 10.20.30.40/32
 no shutdown
!

overlay-gateway g1
 type layer2-extension
 ip interface Loopback 1
 map vni auto
 activate
!
```

EVPN

The following example shows how to configure EVPN.

```
evpn l3gwevpn
 rd auto
 bridge-domain add 100
```

BD VE show commands

This section presents example output of a variety of **show** commands that are useful in BD VE configuration.

show bridge-domain

```
device# show bridge-domain 100
Bridge-domain 100
------------------------------
Bridge-domain Type: MP, VC-ID: 10000 MCT Enabled: FALSE
Number of configured end-points:  1, Number of Active end-points: 1
VE if-indx: 0, Local switching: TRUE, bpdu-drop-enable: TRUE
MAC Withdrawal: Disabled
PW-profile: default, mac-limit: 0
Total VPLS peers: 0 (0 Operational):
```

show tunnel

```
device# show tunnel 61441
Tunnel 61441, mode VXLAN, node-ids 1
Ifindex 0x7c00f001, Admin state up, Oper state up
Overlay gateway "l3vxlangw", ID 1
Source IP 1.1.1.1 (  Loopback 1 ), Vrf default-vrf
Destination IP 2.2.2.2
Configuration source Site
MAC learning enabled
Active next hops on node 1:
    IP: 10.0.1.2, Vrf: default-vrf
    Egress L3 port: Eth 0/5, Outer SMAC: 609c.9f5a.5d1d
    Outer DMAC: 609c.9f5a.551b, ctag: 0
    BUM forwarder: yes

Packet count: RX 2594261         TX 52
Byte count  : RX 306122333       TX 6783
```

show tunnel brief

```
device# show tunnel brief
Tunnel 61441, mode VXLAN, node-ids 1
Admin state up, Oper state up
Source IP 1.1.1.1 (  Loopback 1 ), Vrf default-vrf
Destination IP 2.2.2.2
```

show tunnel status

```
device# show tunnel status
Tnl id    Adm state Oper state BFD state Tnl dest IP     VeId(GRE) Ve state(GRE)
========= ========= ========== ========= =============== ========= ==============
61441     up        up                  100.11.11.11     NA         NA
SLX# show tunnel statistics
Tnl ID   RX packets      TX packets      RX bytes             TX bytes
======== =============== =============== ==================== ====================
61441    2699275         52              318513985            6783
```

show arp

```
device# show arp
Entries in VRF default-vrf : 3
Address          Mac-address     Interface     MacResolved   Age         Type
-----------------------------------------------------------------------------
10.0.1.2         609c.9f5a.551b  Eth 0/5       yes           00:51:47    Dynamic
11.0.1.2         609c.9f5a.5515  Ve 100        yes           00:00:43    Dynamic
92.92.92.2       0000.0ac3.c9ad  Eth 0/4       yes           00:42:12    Dynamic
```

show mac

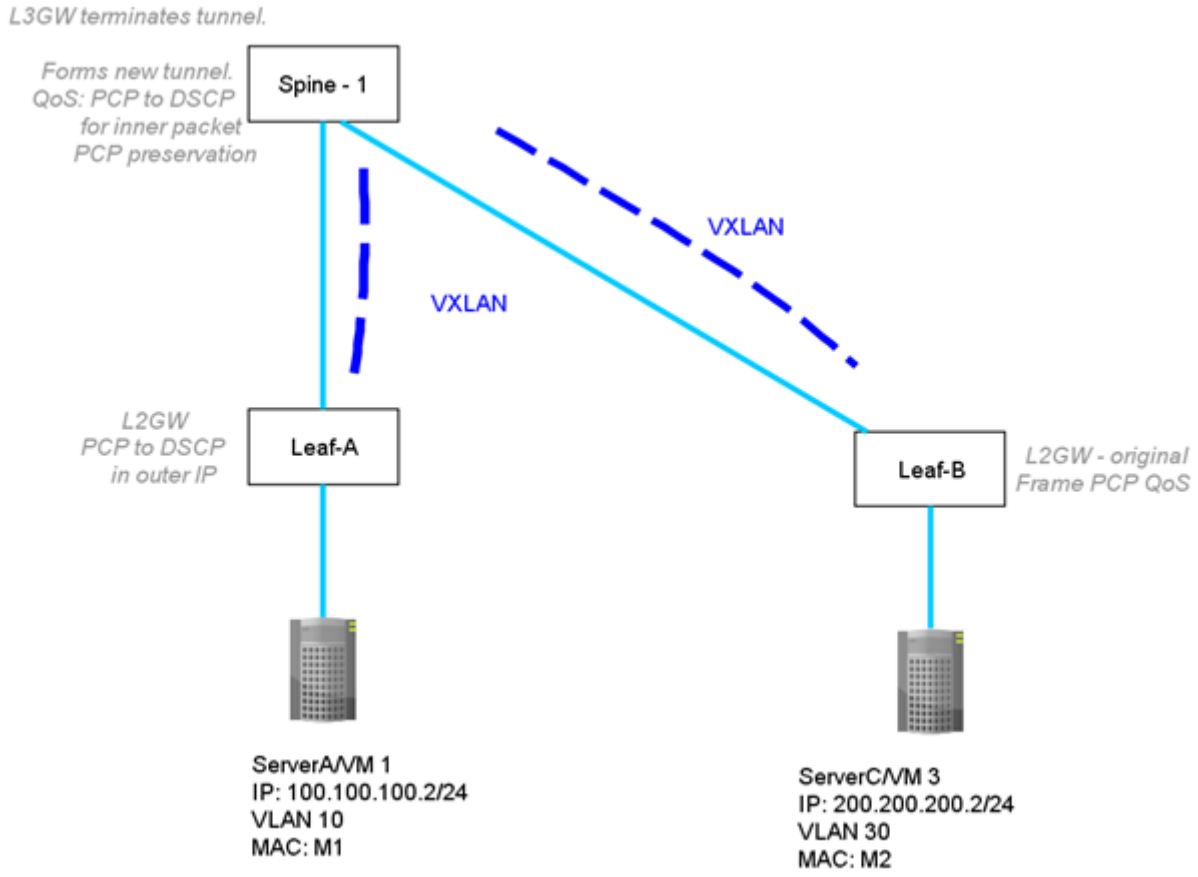
```
device# show mac
VlanId/BDId   Mac-address       Type        State      Ports/LIF/PW
100 (B)       609c.9f5a.5515    Dynamic     Active     tu61441
Total MAC addresses    :  1
```

QoS for VXLAN Layer 2 and Layer 3 gateway interconnections

SLX-OS VXLAN Layer 2 and Layer 3 gateway interconnections can support QoS.

VXLAN Layer 3 gateways resolve the ARP for host VMs and create routes. When VM1 sends intersubnet packets to VM3, Leaf-A sends the packet to Spine - 1 over the VXLAN configuration in the following figure. First, Spine - 1 de-encapsulates the original L2 frame inside the VXLAN tunnel, and the L2 frame has DMAC as the gateway MAC of Spine - 1. Then, Spine - 1 routes the packet to VM3, where the ARP resolution is through the VXLAN configuration.

FIGURE 66 QoS for VXLAN Layer 2 and Layer 3 gateway interconnections



Configuring QoS for VXLAN Layer 2 and Layer 3 gateway interconnections

Enter the **qos-ttl-mode uniform** command to set the QoS type mode to uniform, which is the default, when configuring the VXLAN L2 gateway. For example, enter the following commands when configuring the gateway for the ingress (tunneling) side of packet forwarding:

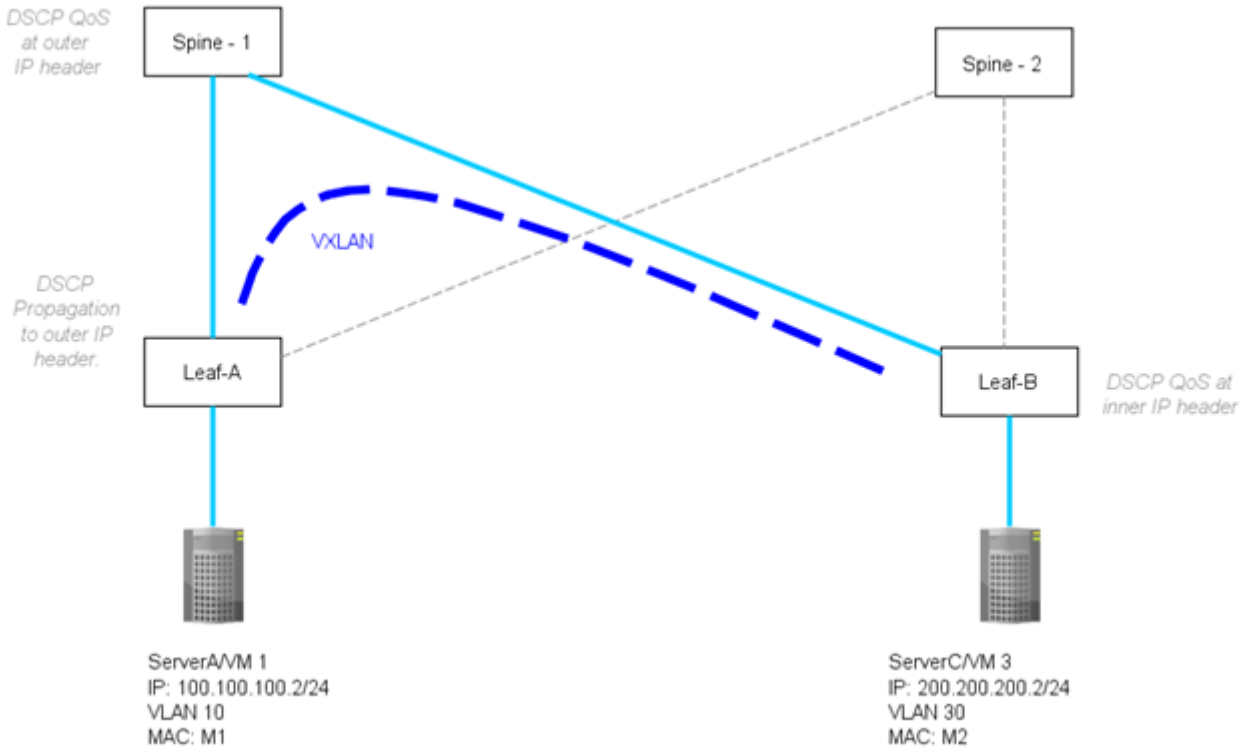
```
device(config)# overlay-gateway gateway_L2L3
device(config-overlay-gw-gateway_L2L3)# type layer2-extension
device(config-overlay-gw-gateway_L2L3)# ip interface loopback 1
device(config-overlay-gw-gateway_L2L3)# qos-ttl-mode uniform
device(config-overlay-gw-gateway_L2L3)# map vni auto
device(config-overlay-gw-gateway_L2L3)# activate
```

QoS for VXLAN Layer 3 gateways

SLX-OS VXLAN Layer 3 gateways can support QoS.

A VXLAN L3 gateway allows intersubnet communication through the VXLAN configuration. In the following figure, the leaf nodes act as the VXLAN L3 gateway while Spine - 1 and Spine - 2 act as IP routing nodes that just route the VXLAN packet out between Leaf-A and Leaf-B. Hosts on different subnets cannot learn MAC addresses of each other in a VXLAN network. The leaf nodes resolve ARP for a tenant VM. The leaf nodes advertise the host routes to each other and tenant VMs with the next hop configured as VTEP. In the following figure, VM1 on a VXLAN network is sending a packet to VM3 on a VXLAN network. At Leaf-A, the DSCP of the tenant packet is propagated to the outer IP header. Leaf-A also adds a L2 header corresponding to the VTEP gateway MAC, after the VXLAN header. Optionally, the 802.1p VLAN/PCP marking of this inner Ethernet header may be set to a value corresponding to the received Layer 2 traffic class of the tenant packet. The Leaf-B terminating IP and L2 headers ignore the QoS marking of transport network and proceed with marking the inner IP header.

FIGURE 67 VXLAN L3 gateway support of intersubnet communication



Configuring QoS for VXLAN Layer 3 gateways

First, complete the required configuration for VXLAN Layer 3 gateways, refer to Configuring TCAM profiles to support Layer 3 gateway on page 452. Then, enter the **qos-ttl-mode uniform** command to set the QoS type mode to uniform, which is the default, when configuring the VXLAN L3 gateway:

```
device(config)# overlay-gateway gateway_L3
device(config-overlay-gw-gateway_L3)# type layer2-extension
device(config-overlay-gw-gateway_L3)# ip interface loopback 1
device(config-overlay-gw-gateway_L3)# qos-ttl-mode uniform
device(config-overlay-gw-gateway_L3)# map vni auto
device(config-overlay-gw-gateway_L3)# activate
```