

Brocade FastIron Layer 3 Routing Configuration Guide, 08.0.50

Supporting Brocade FastIron OS 08.0.50

© 2016, Brocade Communications Systems, Inc. All Rights Reserved.

Brocade, the B-wing symbol, and MyBrocade are registered trademarks of Brocade Communications Systems, Inc., in the United States and in other countries. Other brands, product names, or service names mentioned of Brocade Communications Systems, Inc. are listed at www.brocade.com/en/legal/brocade-Legal-intellectual-property/brocade-legal-trademarks.html. Other marks may belong to third parties.

Notice: This document is for informational purposes only and does not set forth any warranty, expressed or implied, concerning any equipment, equipment feature, or service offered or to be offered by Brocade. Brocade reserves the right to make changes to this document at any time, without notice, and assumes no responsibility for its use. This informational document describes features that may not be currently available. Contact a Brocade sales office for information on feature and product availability. Export of technical data contained in this document may require an export license from the United States government.

The authors and Brocade Communications Systems, Inc. assume no liability or responsibility to any person or entity with respect to the accuracy of this document or any loss, cost, liability, or damages arising from the information contained herein or the computer programs that accompany it.

The product described by this document may contain open source software covered by the GNU General Public License or other open source license agreements. To find out which open source software is included in Brocade products, view the licensing terms applicable to the open source software, and obtain a copy of the programming source code, please visit <http://www.brocade.com/support/oscd>.

Contents

Preface.....	17
Document conventions.....	17
Notes, cautions, and warnings.....	17
Text formatting conventions.....	17
Command syntax conventions.....	18
Brocade resources.....	18
Document feedback.....	18
Contacting Brocade Technical Support.....	19
Brocade customers.....	19
Brocade OEM customers.....	19
About This Document.....	21
Supported hardware and software.....	21
What's new in this document.....	21
ARP - Address Resolution Protocol.....	23
ARP parameter configuration.....	23
How ARP works.....	23
Rate limiting ARP packets.....	24
Changing the ARP aging period.....	25
Enabling proxy ARP.....	25
Creating static ARP entries.....	26
ARP Packet Validation.....	28
Ingress ARP packet priority.....	28
Displaying the ARP table	29
Reverse Address Resolution Protocol configuration.....	29
How RARP Differs from BootP and DHCP.....	29
Disabling RARP.....	30
Creating static RARP entries.....	30
Changing the maximum number of static RARP entries supported.....	30
Dynamic ARP inspection	31
ARP poisoning.....	31
Dynamic ARP Inspection.....	31
Configuration notes and feature limitations for DAI.....	32
Dynamic ARP Inspection configuration.....	33
Multi-VRF support for DAI.....	34
Displaying ARP inspection status and ports.....	35
IP Addressing.....	37
IP configuration overview.....	37
Full Layer 3 support.....	37
IP interfaces.....	38
IP packet flow through a Layer 3 switch.....	39
IP route exchange protocols.....	42
IP multicast protocols.....	43
IP interface redundancy protocols.....	43
ACLs and IP access policies.....	43
Basic IP parameters and defaults - Layer 3 switches.....	44

When parameter changes take effect.....	44
IP global parameters - Layer 3 switches.....	44
IP interface parameters - Layer 3 switches.....	48
Basic IP parameters and defaults - Layer 2 switches.....	49
IP global parameters - Layer 2 switches.....	49
Interface IP parameters - Layer 2 switches.....	51
Basic IP configuration.....	51
IP address replacement.....	51
Limitations and restrictions.....	52
Replacing an IP address.....	52
Configuring IP parameters - Layer 3 switches.....	52
Configuring IP addresses.....	53
Configuring 31-bit subnet masks on point-to-point networks.....	56
Configuring DNS resolver.....	58
Configuring packet parameters.....	60
Changing the router ID.....	63
Specifying a single source interface for specified packet types.....	64
Configuring delay time for notifying VE down event.....	67
Configuring forwarding parameters.....	68
Disabling ICMP messages.....	70
Enabling ICMP redirect messages.....	71
Configuring a default network route.....	72
Configuring IP load sharing.....	73
ECMP load sharing for IPv6.....	76
ICMP Router Discovery Protocol configuration.....	78
IRDP parameters.....	78
Configuring UDP broadcast and IP helper parameters.....	80
Configuring IP parameters - Layer 2 switches.....	82
Configuring the management IP address and specifying the default gateway.....	82
Configuring Domain Name System resolver.....	83
Changing the TTL threshold.....	84
IPv4 point-to-point GRE tunnels	85
IPv4 GRE tunnel overview.....	85
GRE packet structure and header format.....	86
Path MTU Discovery support.....	86
Support for IPv4 multicast routing over GRE tunnels.....	87
Configuration considerations for GRE IP tunnels.....	87
Configuration tasks for GRE tunnels.....	88
Example point-to-point GRE tunnel configuration.....	95
Displaying GRE tunneling information.....	96
Clearing GRE statistics.....	100
Bandwidth for IP interfaces.....	100
Limitations and pre-requisites.....	101
OSPF cost calculation with interface bandwidth.....	101
Setting the bandwidth value for an Ethernet interface.....	101
Setting the bandwidth value for a VE interface.....	102
Setting the bandwidth value for a tunnel interface.....	103
User-configurable MAC address per IP interface.....	103
Manually configuring an IP MAC address.....	104
Modifying and displaying Layer 3 system parameter limits.....	105

Layer 3 configuration notes.....	105
Displaying Layer 3 system parameter limits.....	105
Enabling or disabling routing protocols.....	106
Enabling or disabling Layer 2 switching.....	107
Configuration notes and feature limitations for Layer 2 switching.....	107
Command syntax for Layer 2 switching.....	107
Configuring a Layer 3 Link Aggregation Group (LAG).....	107
Disabling IP checksum check.....	108
Displaying IP configuration information and statistics.....	109
Displaying IP information - Layer 3 switches.....	109
Displaying IP information - Layer 2 switches.....	120
IPv6 Addressing.....	125
IPv6 addressing overview.....	125
IPv6 address types.....	126
IPv6 stateless auto-configuration.....	127
Full Layer 3 IPv6 feature support.....	128
IPv6 CLI command support	128
IPv6 host address on a Layer 2 switch.....	130
Configuring a global or site-local IPv6 address with a manually configured interface ID.....	131
Configuring a link-local IPv6 address as a system-wide address for a switch.....	131
Configuring the management port for an IPv6 automatic address configuration.....	132
Configuring basic IPv6 connectivity on a Layer 3 switch.....	132
Enabling IPv6 routing.....	132
IPv6 configuration on each router interface.....	132
Configuring IPv4 and IPv6 protocol stacks.....	135
IPv6 over IPv4 tunnels.....	136
IPv6 over IPv4 tunnel configuration notes.....	136
Configuring a manual IPv6 tunnel.....	137
Clearing IPv6 tunnel statistics.....	137
Displaying IPv6 tunnel information.....	138
Displaying a summary of tunnel information.....	138
Displaying interface level IPv6 settings.....	138
IPv6 management (IPv6 host support).....	139
Configuring IPv6 management ACLs.....	139
Restricting SNMP access to an IPv6 node.....	140
Specifying an IPv6 SNMP trap receiver.....	140
Configuring SNMP V3 over IPv6.....	140
Secure Shell, SCP, and IPv6.....	140
IPv6 Telnet.....	140
IPv6 traceroute.....	141
IPv6 Web management using HTTP and HTTPS.....	141
Restricting Web management access.....	142
Restricting Web management access by specifying an IPv6 ACL.....	142
Restricting Web management access to an IPv6 host.....	142
Configuring name-to-IPv6 address resolution using IPv6 DNS resolver.....	142
Defining an IPv6 DNS entry.....	142
Pinging an IPv6 address.....	143
Configuring an IPv6 Syslog server.....	144
Viewing IPv6 SNMP server addresses.....	144
Disabling router advertisement and solicitation messages.....	145

Disabling IPv6 on a Layer 2 switch.....	145
IPv6 ICMP feature configuration.....	145
Configuring ICMP rate limiting.....	146
Enabling IPv6 ICMP redirect messages.....	146
IPv6 neighbor discovery configuration.....	147
IPv6 neighbor discovery configuration notes.....	147
Neighbor solicitation and advertisement messages.....	147
Router advertisement and solicitation messages.....	148
Neighbor redirect messages.....	148
Setting neighbor solicitation parameters for duplicate address detection.....	148
Setting IPv6 router advertisement parameters.....	149
Prefixes advertised in IPv6 router advertisement messages.....	150
Setting flags in IPv6 router advertisement messages.....	151
Enabling and disabling IPv6 router advertisements.....	152
IPv6 router advertisement preference support.....	152
Configuring reachable time for remote IPv6 nodes.....	152
IPv6 neighbor discovery inspection.....	153
Neighbor discovery inspection configuration.....	156
Syslog message for ND inspection.....	156
IPv6 MTU.....	156
Configuration notes and feature limitations for IPv6 MTU.....	156
Changing the IPv6 MTU.....	157
Static neighbor entries configuration.....	157
Limiting the number of hops an IPv6 packet can traverse.....	158
IPv6 source routing security enhancements.....	158
TCAM space configuration.....	158
Allocating TCAM space.....	159
Allocating TCAM space for GRE tunnels.....	160
Displaying global IPv6 information.....	161
Displaying IPv6 cache information.....	161
Displaying IPv6 interface information.....	162
Displaying IPv6 neighbor information.....	163
Displaying the IPv6 route table	164
Displaying local IPv6 routers.....	166
Displaying IPv6 TCP information.....	166
Displaying IPv6 traffic statistics.....	169
Clearing global IPv6 information.....	172
Clearing the IPv6 cache.....	172
Clearing IPv6 neighbor information.....	172
Clearing IPv6 routes from the IPv6 route table.....	173
Clearing IPv6 traffic statistics.....	173
IPv4 Static Routing.....	175
Overview of static routing.....	175
Static route states follow port states.....	176
Configuring a basic IP static route.....	176
Adding metrics to a static route.....	177
Naming an IP static route.....	178
Removing a name or a static route.....	178
Configuring a physical interface as next hop.....	179
Configuring a virtual interface as next hop.....	179

Configuring a tunnel as next hop.....	180
Configuring a static route for use with a route map.....	180
Configuring a null route.....	180
Configuring a default static route.....	182
Resolving a static route using other static routes.....	182
Resolving the next hop through a protocol.....	183
Creating an IP static route in a non-default VRF.....	183
Configuring load sharing and redundancy.....	184
Determining maximum static routes.....	186
Displaying IPv4 static routes.....	187
IPv6 Static Routing.....	189
Overview of static routing.....	189
Static route states follow port states.....	190
Configuring a basic IPv6 static route.....	190
Removing an IPv6 static route.....	191
Configuring an interface as next hop.....	191
Configuring a virtual interface as next hop.....	192
Configuring a tunnel as next hop.....	193
Configuring a VRF as next hop for an IPv6 static route.....	193
Adding metrics to an IPv6 static route.....	193
Configuring a null route.....	194
Configuring a default static route.....	196
Resolving a static route using other static routes.....	196
Resolving the IPv6 static route through a protocol.....	197
Configuring load sharing and redundancy.....	197
Adding an IPv6 static route tag for use with route-maps.....	199
IPv6 multicast static routes.....	199
Configuring IPv6 multicast routes in a non-default VRF.....	199
Determining maximum static routes.....	201
Displaying information on IPv6 static routes.....	203
RIP.....	205
RIP overview.....	205
RIP parameters and defaults.....	205
RIP global parameters.....	205
RIP interface parameters.....	207
Configuring RIP parameters.....	207
Enabling RIP.....	207
Configuring route costs.....	208
Changing the administrative distance.....	208
Configuring redistribution.....	208
Configuring route learning and advertising parameters.....	210
Changing the route loop prevention method.....	211
Suppressing RIP route advertisement on a VRRP or VRRPE backup interface.....	212
Configuring RIP route filters using prefix-lists and route maps.....	212
Setting RIP timers.....	214
Displaying RIP Information.....	214
Displaying CPU utilization statistics.....	216
RIPng.....	219
RIPng Overview.....	219

Configuring RIPng.....	219
Enabling RIPng.....	219
Configuring RIPng timers.....	220
Configuring route learning and advertising parameters.....	221
Redistributing routes into RIPng.....	222
Controlling distribution of routes through RIPng.....	223
Configuring poison reverse parameters.....	223
Clearing RIPng routes from IPv6 route table.....	224
Displaying RIPng configuration.....	224
Displaying RIPng routing table.....	225
OSPFv2.....	227
OSPFv2 overview.....	228
Autonomous System.....	228
OSPFv2 components and roles.....	229
Area Border Routers.....	229
Autonomous System Boundary Routers.....	229
Designated routers.....	229
Reduction of equivalent AS external LSAs.....	230
Algorithm for AS external LSA reduction.....	232
Enabling OSPFv2.....	232
Backbone area.....	232
Assigning OSPFv2 areas.....	233
Area range.....	233
Assigning an area range.....	233
Area types.....	234
Stub area and totally stubby area.....	234
Disabling summary LSAs for a stub area.....	235
Not-so-stubby area (NSSA).....	235
Configuring an NSSA.....	236
Configuring a summary-address for the NSSA.....	237
Assigning interfaces to an area.....	237
Link state advertisements.....	238
Virtual links.....	238
Configuring virtual links.....	239
Default route origination.....	240
External route summarization.....	241
SPF timers.....	241
Modifying Shortest Path First timers.....	242
OSPFv2 LSA refreshes.....	242
Configuring the OSPFv2 LSA pacing interval.....	243
Support for OSPF RFC 2328 Appendix E.....	243
OSPFv2 graceful restart.....	244
Disabling OSPFv2 graceful restart.....	244
Re-enabling OSPFv2 graceful restart.....	245
Disabling OSPFv2 graceful restart helper.....	245
OSPFv2 stub router advertisement.....	246
OSPFv2 Shortest Path First throttling.....	246
IETF RFC and internet draft support.....	247
OSPFv2 non-stop routing.....	247
Limitations of NSR.....	247

Enabling OSPFv2 NSR.....	248
Synchronization of critical OSPFv2 elements.....	248
Link state database synchronization.....	248
LSA delayed acknowledging.....	248
LSA syncing and packing	249
Neighbor device synchronization.....	249
Synchronization limitations.....	249
Interface synchronization.....	249
Standby module operations.....	249
Neighbor database.....	250
LSA database.....	250
OSPFv2 distribute list.....	250
Configuring an OSPFv2 distribution list using ACLs	250
Configuring an OSPFv2 distribution list using route maps	251
OSPFv2 route redistribution.....	252
Redistributing routes into OSPFv2.....	253
Load sharing.....	254
Interface types to which the reference bandwidth does not apply.....	255
Changing the reference bandwidth for the cost on OSPFv2 interfaces.....	255
OSPFv2 over VRF.....	256
Enabling OSPFv2 in a non-default VRF.....	256
Configuring the OSPFv2 Max-Metric Router LSA.....	257
Re-enabling OSPFv2 compatibility with RFC 1583.....	257
Changing default settings.....	257
Disabling and re-enabling OSPFv2 event logging.....	258
Understanding the effects of disabling OSPFv2.....	258
Disabling OSPFv2.....	258
Displaying OSPFv2 results.....	259
OSPFv3.....	263
OSPFv3 overview.....	263
Configuring the router ID.....	264
Enabling OSPFv3.....	264
Configuring OSPFv3.....	264
OSPFv3 areas.....	265
Backbone area.....	265
Area range.....	265
Area types.....	265
Assigning OSPFv3 areas.....	266
Assigning OSPFv3 areas to interfaces.....	266
Stub area and totally stubby area.....	267
Assigning a stub area.....	268
Not-so-stubby area.....	268
Configuring an NSSA.....	269
LSA types for OSPFv3.....	269
Virtual links.....	270
Virtual link source address assignment.....	272
Configuring virtual links.....	272
OSPFv3 route redistribution.....	273
Redistributing routes into OSPFv3.....	274
Default route origination.....	275

Configuring default external routes.....	275
Disabling and re-enabling OSPFv3 event logging.....	275
Filtering OSPFv3 routes.....	276
Configuring an OSPFv3 distribution list using an IPv6 prefix list as input.....	276
Configuring an OSPFv3 distribution list using a route map as input.....	278
SPF timers.....	279
Modifying SPF timers.....	280
OSPFv3 administrative distance.....	280
Configuring administrative distance based on route type.....	280
Changing the reference bandwidth for the cost on OSPFv3 interfaces.....	281
OSPFv3 LSA refreshes.....	282
Configuring the OSPFv3 LSA pacing interval.....	282
External route summarization.....	283
OSPFv3 over VRF.....	283
Enabling OSPFv3 in a non-default VRF.....	284
Assigning OSPFv3 areas in a non-default VRF.....	284
Setting all OSPFv3 interfaces to the passive state.....	285
OSPFv3 graceful restart helper.....	286
Disabling OSPFv3 graceful restart helper.....	286
Re-enabling OSPFv3 graceful restart helper.....	286
OSPFv3 non-stop routing.....	287
Enabling OSPFv3 NSR.....	287
IPsec for OSPFv3.....	288
IPsec for OSPFv3 configuration.....	288
IPsec for OSPFv3 considerations.....	289
Configuring IPsec on an OSPFv3 area.....	289
Configuring IPsec on an OSPFv3 interface.....	290
Configuring IPsec on OSPFv3 virtual links.....	291
Specifying the key rollover timer.....	292
Clearing IPsec statistics.....	292
Displaying OSPFv3 results.....	293
BGP4.....	297
BGP4 overview.....	298
Neighbor redirect messages.....	298
Relationship between the BGP4 route table and the IP route table.....	299
How BGP4 selects a path for a route (BGP best path selection algorithm).....	299
BGP4 message types.....	301
Grouping of RIB-out peers.....	302
Implementation of BGP4.....	302
BGP4 restart.....	303
BGP4 Peer notification during a management module switchover.....	303
BGP4 neighbor local AS.....	305
Basic configuration and activation for BGP4.....	306
BGP4 parameters.....	307
Memory considerations.....	308
Memory configuration options obsoleted by dynamic memory.....	308
Enabling BGP4 on the device.....	309
Changing the device ID.....	309
Setting the local AS number.....	309
Setting the local AS number for VRF instances.....	310

Adding a loopback interface.....	311
Adding BGP4 neighbors.....	311
Removing route dampening from suppressed routes.....	315
Encrypting BGP4 MD5 authentication keys.....	316
Displaying neighbor information.....	318
Clearing IPv6 route information.....	318
Adding a BGP4 peer group.....	319
Disabling BGP4.....	321
Changing the Keep Alive Time and Hold Time.....	322
Changing the BGP4 next-hop update timer.....	322
Enabling fast external fallover.....	323
Changing the maximum number of paths for BGP4 Multipath load sharing.....	323
How Multipath load sharing affects route selection.....	324
How Multipath load sharing works.....	324
Changing the maximum number of shared BGP4 paths.....	324
Customizing BGP4 Multipath load sharing.....	325
Enhancements to BGP4 Multipath load sharing	325
Specifying a list of networks to advertise.....	326
Specifying a route map when configuring BGP4 network advertising.....	326
Changing the default local preference.....	326
Using the IP default route as a valid next-hop for a BGP4 route.....	327
Changing the default MED (Metric) used for route redistribution.....	327
Enabling next-hop recursion.....	328
Enabling recursive next-hop lookups.....	328
Example when recursive route lookups are disabled.....	329
Example when recursive route lookups are enabled.....	329
Changing administrative distances.....	330
Requiring the first AS to be the neighbor AS.....	331
Disabling or re-enabling comparison of the AS-Path length.....	332
Enabling or disabling comparison of device IDs.....	332
Configuring the device to always compare Multi-Exit Discriminators.....	333
Treating missing MEDs as the worst MEDs.....	334
Configuring route reflection parameters.....	334
Support for RFC 4456.....	335
Configuration procedures for BGP4 route reflector.....	336
Disabling or re-enabling client-to-client route reflection.....	336
Configuring confederations.....	336
Configuring a BGP4 confederation.....	338
Aggregating routes advertised to BGP4 neighbors.....	339
Configuring BGP4 restart.....	340
Configuring BGP4 Restart for the global routing instance.....	340
Configuring BGP4 Restart for a VRF.....	340
Configuring timers for BGP4 Restart (optional).....	340
BGP4 null0 routing.....	341
Configuring BGP4 null0 routing.....	342
Modifying redistribution parameters.....	345
Redistributing connected routes.....	345
Redistributing RIP routes.....	346
Redistributing OSPF external routes.....	346
Redistributing static routes.....	347

Redistributing IBGP routes.....	347
Filtering.....	347
AS-path filtering.....	347
BGP4 filtering communities.....	350
Defining and applying IP prefix lists.....	351
Defining neighbor distribute lists.....	352
Defining route maps.....	353
Using a table map to set the tag value.....	360
Cooperative BGP4 route filtering.....	361
Four-byte Autonomous System Numbers (AS4).....	363
Enabling AS4 numbers.....	364
BGP4 AS4 attribute errors.....	368
Error logs.....	368
Configuring route flap dampening.....	368
Globally configuring route flap dampening.....	369
Using a route map to configure route flap dampening for a specific neighbor.....	370
Removing route dampening from a route.....	371
Displaying and clearing route flap dampening statistics.....	371
Generating traps for BGP4.....	372
Configuring BGP4.....	373
Entering and exiting the address family configuration level.....	374
BGP route reflector.....	374
Configuring BGP route reflector.....	375
Specifying a maximum AS path length.....	377
Setting a global maximum AS path limit.....	378
Setting a maximum AS path limit for a peer group or neighbor.....	378
BGP4 max-as error messages.....	379
Maximum AS path limit error.....	379
Memory limit error.....	379
Originating the default route.....	379
Changing the default metric used for route cost.....	379
Configuring a static BGP4 network	380
Setting an administrative distance for a static BGP4 network.....	380
Limiting advertisement of a static BGP4 network to selected neighbors.....	381
Dynamic route filter update.....	381
Generalized TTL Security Mechanism support.....	383
Displaying BGP4 information.....	383
Displaying summary BGP4 information.....	383
Displaying the active BGP4 configuration.....	386
Displaying summary neighbor information.....	386
Displaying BGP4 neighbor information.....	388
Displaying peer group information.....	396
Displaying summary route information.....	396
Displaying VRF instance information.....	397
Displaying the BGP4 route table.....	397
Displaying BGP4 route-attribute entries.....	404
Displaying the routes BGP4 has placed in the IP route table.....	405
Displaying route flap dampening statistics.....	405
Displaying the active route map configuration.....	406
Displaying BGP4 graceful restart neighbor information.....	407

Displaying AS4 details.....	407
Updating route information and resetting a neighbor session.....	414
Using soft reconfiguration.....	414
Dynamically requesting a route refresh from a BGP4 neighbor.....	416
Closing or resetting a neighbor session.....	418
Clearing and resetting BGP4 routes in the IP route table.....	419
Clearing traffic counters.....	419
BGP4+.....	421
BGP4+ overview.....	421
BGP global mode	421
IPv6 unicast address family.....	423
BGP4+ neighbors.....	424
Configuring BGP4+ neighbors using global IPv6 addresses.....	424
Configuring BGP4+ neighbors using link-local addresses.....	425
BGP4+ peer groups.....	426
Configuring BGP4+ peer groups.....	426
Configuring a peer group with IPv4 and IPv6 peers.....	427
Importing routes into BGP4+.....	428
Advertising the default BGP4+ route.....	429
Advertising the default BGP4+ route to a specific neighbor.....	429
Using the IPv6 default route as a valid next hop for a BGP4+ route.....	430
BGP4+ next hop recursion.....	430
Enabling next-hop recursion.....	431
BGP4+ NLRIs and next hop attributes.....	431
BGP4+ route reflection.....	432
Configuring a cluster ID for a route reflector.....	432
Configuring a route reflector client.....	433
BGP4+ route aggregation.....	433
Aggregating routes advertised to BGP neighbors.....	433
BGP4+ multipath.....	434
Enabling load-balancing across different paths.....	435
Route maps.....	435
Configuring a route map for BGP4+ prefixes.....	436
Redistributing prefixes into BGP4+.....	437
Redistributing routes into BGP4+.....	437
Specifying the weight added to BGP4+ received routes.....	438
BGP4+ outbound route filtering.....	438
Configuring BGP4+ outbound route filtering.....	439
BGP4+ confederations.....	440
Configuring BGP4+ confederations.....	440
BGP4+ extended community.....	441
Defining a community ACL.....	441
Applying a BGP extended community filter.....	442
BGP4+ graceful restart.....	443
Disabling BGP4+ graceful restart.....	444
Re-enabling BGP4+ graceful restart.....	444
Generalized TTL Security Mechanism support.....	446
Assumptions and limitations.....	446
Configuring GTSM for BGP4+.....	446
Disabling the BGP AS_PATH check function.....	447

Displaying BGP4+ statistics.....	448
Displaying BGP4+ neighbor statistics.....	450
VRRPv2.....	453
VRRPv2 overview.....	453
VRRP terminology.....	455
VRRP hold timer.....	456
VRRP interval timers.....	456
VRRP authentication.....	457
VRRP master device abdication to backup device.....	458
ARP and VRRP control packets.....	458
Enabling an owner VRRP device.....	458
Enabling a backup VRRP device.....	460
Configuring simple text authentication on VRRP interfaces.....	461
Configuring MD5 authentication on VRRP interfaces.....	462
Abdicating VRRP master device status.....	463
Tracked ports and track priority with VRRP and VRRP-E.....	465
Tracking ports and setting the VRRP priority.....	465
VRRP backup preemption.....	466
Disabling VRRP backup preemption.....	466
Accept mode for backup VRRP devices.....	467
Enabling accept mode on a backup VRRP device.....	467
Suppressing RIP route advertisements on VRRP backup devices.....	469
VRRP-Ev2 overview.....	469
Enabling a VRRP-E device.....	470
VRRP-E load-balancing using short-path forwarding.....	471
Packet routing with short-path forwarding to balance traffic load.....	471
Short-path forwarding with revert priority.....	472
Configuring VRRP-E load-balancing using short-path forwarding.....	473
VRRP-E hitless upgrade.....	474
Configuring VRRP-E hitless upgrade.....	474
VRRP-E slow start timer.....	476
Configuring a VRRP-E slow-start timer.....	476
Configuration example: ISSU upgrade using VRRP-E.....	477
Displaying VRRPv2 information.....	479
Clearing VRRPv2 statistics.....	480
VRRPv3.....	483
VRRPv3 overview.....	483
Enabling an IPv6 VRRPv3 owner device.....	484
Enabling an IPv6 VRRPv3 backup device.....	485
Enabling an IPv4 VRRPv3 owner device.....	486
Enabling an IPv4 VRRPv3 backup device.....	487
Tracked ports and track priority with VRRP and VRRP-E.....	488
Tracking ports and setting VRRP priority using VRRPv3.....	489
Accept mode for backup VRRP devices.....	489
Enabling accept mode on a backup VRRP device.....	490
Alternate VRRPv2 checksum for VRRPv3 IPv4 sessions.....	491
Enabling the VRRPv2 checksum computation method in a VRRPv3 IPv4 session.....	491
Displaying alternate VRRPv2 checksum settings.....	492
Automatic generation of a virtual link-local address for VRRPv3.....	493

Assigning an auto-generated link-local IPv6 address for a VRRPv3 cluster.....	494
Displaying VRRPv3 statistics.....	495
Clearing VRRPv3 statistics.....	496
VRRP-Ev3 Overview.....	496
Enabling an IPv6 VRRP-Ev3 device.....	497
Displaying and clearing VRRP-Ev3 statistics.....	498
Multi-VRF.....	501
Multi-VRF overview.....	501
FastIron considerations for Multi-VRF.....	503
VRF-related system-max values.....	503
Additional features to support Multi-VRF.....	505
Configuring Multi-VRF.....	506
Configuring VRF system-max values	506
Creating VLANs as links on a tagged port for security.....	508
Configuring a VRF instance.....	508
Starting a routing process for a VRF.....	509
Assigning a Layer 3 interface to a VRF.....	510
Assigning a loopback interface to a VRF.....	510
Verifying a Multi-VRF configuration.....	511
Removing a VRF configuration.....	512
Configuring static ARP for Multi-VRF.....	512
Configuring additional ARP features for Multi-VRF.....	513

Preface

• Document conventions.....	17
• Brocade resources.....	18
• Document feedback.....	18
• Contacting Brocade Technical Support.....	19

Document conventions

The document conventions describe text formatting conventions, command syntax conventions, and important notice formats used in Brocade technical documentation.

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.
<i>italic text</i>	Identifies emphasis.
	Identifies variables.
	Identifies document titles.
Courier font	Identifies CLI output.

Format	Description
	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.
<i>italic text</i>	Identifies a variable.
value	In Fibre Channel products, a fixed value provided as input to a command option is printed in plain text, for example, --show WWN.
[]	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. In Fibre Channel products, square brackets may be used instead for this purpose.
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, <i>member[member...]</i> .
\	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.

Brocade resources

Visit the Brocade website to locate related documentation for your product and additional Brocade resources.

White papers, data sheets, and the most recent versions of Brocade software and hardware manuals are available at www.brocade.com.

Product documentation for all supported releases is available to registered users at MyBrocade.

Click the **Support** tab and select **Document Library** to access documentation on MyBrocade or www.brocade.com. You can locate documentation by product or by operating system.

Release notes are bundled with software downloads on MyBrocade. Links to software downloads are available on the MyBrocade landing page and in the Document Library.

Document feedback

Quality is our first concern at Brocade, and we have made every effort to ensure the accuracy and completeness of this document. However, if you find an error or an omission, or you think that a topic needs further development, we want to hear from you. You can provide feedback in two ways:

- Through the online feedback form in the HTML documents posted on www.brocade.com
- By sending your feedback to documentation@brocade.com

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

Contacting Brocade Technical Support

As a Brocade customer, you can contact Brocade Technical Support 24x7 online, by telephone, or by e-mail. Brocade OEM customers should contact their OEM/solution provider.

Brocade customers

For product support information and the latest information on contacting the Technical Assistance Center, go to www.brocade.com and select **Support**.

If you have purchased Brocade product support directly from Brocade, use one of the following methods to contact the Brocade Technical Assistance Center 24x7.

Online	Telephone	E-mail
<p>Preferred method of contact for non-urgent issues:</p> <ul style="list-style-type: none"> Case management through the MyBrocade portal. Quick Access links to Knowledge Base, Community, Document Library, Software Downloads and Licensing tools 	<p>Required for Sev 1-Critical and Sev 2-High issues:</p> <ul style="list-style-type: none"> Continental US: 1-800-752-8061 Europe, Middle East, Africa, and Asia Pacific: +800-AT FIBREE (+800 28 34 27 33) Toll-free numbers are available in many countries. For areas unable to access a toll-free number: +1-408-333-6061 	<p>support@brocade.com</p> <p>Please include:</p> <ul style="list-style-type: none"> Problem summary Serial number Installation details Environment description

Brocade OEM customers

If you have purchased Brocade product support from a Brocade OEM/solution provider, contact your OEM/solution provider for all of your product support needs.

- OEM/solution providers are trained and certified by Brocade to support Brocade® products.
- Brocade provides backline support for issues that cannot be resolved by the OEM/solution provider.
- Brocade Supplemental Support augments your existing OEM support contract, providing direct access to Brocade expertise. For more information, contact Brocade or your OEM.
- For questions regarding service levels and response times, contact your OEM/solution provider.

About This Document

- [Supported hardware and software](#).....21
- [What's new in this document](#).....21

Supported hardware and software

This guide supports the following product families for FastIron release 08.0.50:

- Brocade ICX 7250 Series (ICX 7250)
- Brocade ICX 7450 Series (ICX 7450)
- Brocade ICX 7750 Series (ICX 7750)

For information about the specific models and modules supported in a product family, refer to the hardware installation guide for that product family.

What's new in this document

The following tables describe information added or modified in this guide for FastIron software release 08.0.50.

TABLE 1 Summary of enhancements in FastIron release 08.0.50

Feature	Description	Location
IP Address Replacement	IP Address replacement allows change management of an IP address across many interfaces in one step.	IP address replacement on page 51
Multi-VRF	Updates to system-max values for IPv6 routes on Brocade ICX 7250; minor updates to configuration	VRF-related system-max values on page 503
Route only	Route only provides the ability to enable or disable Layer 2 switching.	Enabling or disabling Layer 2 switching on page 107
VRRP-E hitless upgrade	When enabled, VRRP-E hitless upgrade decreases the priority of the master router VRID to allow upgrades or troubleshooting on the master router. When combined with the short-path forwarding functionality, traffic loss can be avoided.	VRRP-E hitless upgrade on page 474

ARP - Address Resolution Protocol

• ARP parameter configuration.....	23
• Displaying the ARP table	29
• Reverse Address Resolution Protocol configuration.....	29
• Dynamic ARP inspection	31

ARP parameter configuration

Address Resolution Protocol (ARP) is a standard IP protocol that enables an IP Layer 3 switch to obtain the MAC address of another device interface when the Layer 3 switch knows the IP address of the interface. ARP is enabled by default and cannot be disabled.

NOTE

Brocade Layer 2 switches also support ARP. However, the configuration options described later in this section apply only to Layer 3 switches, not to Layer 2 switches.

How ARP works

A Layer 3 switch needs to know a destination MAC address when forwarding traffic, because the Layer 3 switch encapsulates the IP packet in a Layer 2 packet (MAC layer packet) and sends the Layer 2 packet to a MAC interface on a device directly attached to the Layer 3 switch. The device can be the packet final destination or the next-hop router toward the destination.

The Layer 3 switch encapsulates IP packets in Layer 2 packets regardless of whether the ultimate destination is locally attached or is multiple router hops away. Because the Layer 3 switch IP route table and IP forwarding cache contain IP address information but not MAC address information, the Layer 3 switch cannot forward IP packets based solely on the information in the route table or forwarding cache. The Layer 3 switch needs to know the MAC address that corresponds with the IP address of either the packet locally attached destination or the next-hop router that leads to the destination.

For example, to forward a packet whose destination is multiple router hops away, the Layer 3 switch must send the packet to the next-hop router toward its destination, or to a default route or default network route if the IP route table does not contain a route to the packet destination. In each case, the Layer 3 switch must encapsulate the packet and address it to the MAC address of a locally attached device, the next-hop router toward the IP packet destination.

To obtain the MAC address required for forwarding a datagram, the Layer 3 switch first looks in the ARP cache (not the static ARP table) for an entry that lists the MAC address for the IP address. The ARP cache maps IP addresses to MAC addresses. The cache also lists the port attached to the device and, if the entry is dynamic, the age of the entry. A dynamic ARP entry enters the cache when the Layer 3 switch receives an ARP reply or receives an ARP request (which contains the sender IP address and MAC address). A static entry enters the ARP cache from the separate static ARP table when the interface for the entry comes up.

To ensure the accuracy of the ARP cache, each dynamic entry has its own age timer. The timer is reset to zero each time the Layer 3 switch receives an ARP reply or ARP request containing the IP address and MAC address of the entry. If a dynamic entry reaches its maximum allowable age, the entry times out and the software removes the entry from the table. Static entries do not age out and can be removed only by you.

If the ARP cache does not contain an entry for the destination IP address, the Layer 3 switch broadcasts an ARP request out all its IP interfaces. The ARP request contains the IP address of the destination. If the device with the IP address is directly attached to the Layer 3 switch, the device sends an ARP response containing its MAC address. The response is a unicast packet addressed directly to the Layer 3 switch. The Layer 3 switch places the information from the ARP response into the ARP cache.

ARP requests contain the IP address and MAC address of the sender, so all devices that receive the request learn the MAC address and IP address of the sender and can update their own ARP caches accordingly.

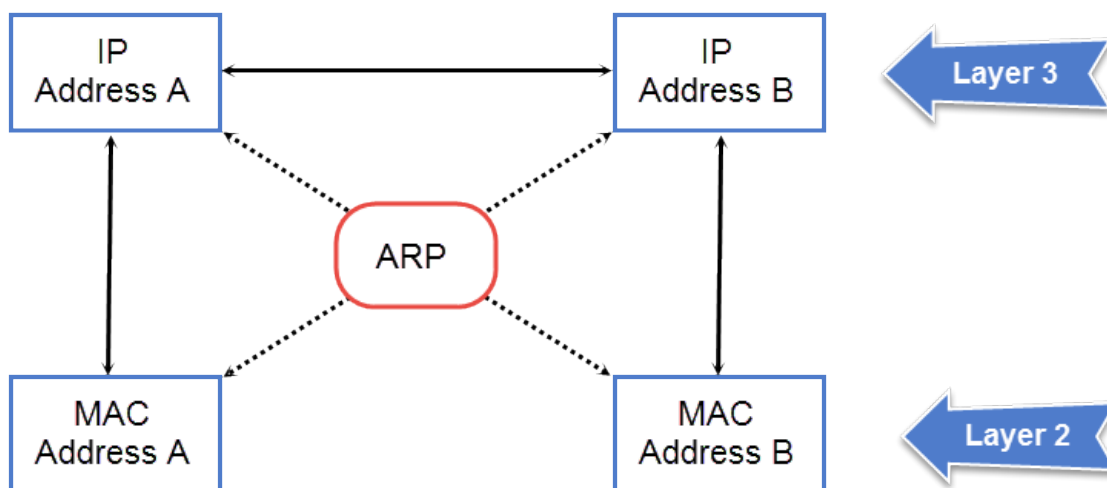
NOTE

The ARP request broadcast is a MAC broadcast, which means the broadcast goes only to devices that are directly attached to the Layer 3 switch. A MAC broadcast is not routed to other networks. However, some routers, including Brocade Layer 3 switches, can be configured to reply to ARP requests from one network on behalf of devices on another network.

NOTE

If the router receives an ARP request packet that it is unable to deliver to the final destination because of the ARP timeout and no ARP response is received (the Layer 3 switch knows of no route to the destination address), the router sends an ICMP Host Unreachable message to the source.

FIGURE 1 ARP supplies the MAC address corresponding to an IP address



If Device A wants to communicate with Device B, knowing the IP address of Device B is not sufficient; the MAC address is also required. ARP supplies the MAC address.

Rate limiting ARP packets

You can limit the number of ARP packets the Brocade device accepts during each second. By default, the software does not limit the number of ARP packets the device can receive. Since the device sends ARP packets to the CPU for processing, if a device in a busy network receives a high number of ARP packets in a short period of time, some CPU processing might be deferred while the CPU processes the ARP packets.

To prevent the CPU from becoming flooded by ARP packets in a busy network, you can restrict the number of ARP packets the device will accept each second. When you configure an ARP rate limit, the device accepts up to the maximum number of packets you specify, but drops additional ARP packets received during the one-second interval. When a new one-second interval starts, the counter restarts at zero, so the device again accepts up to the maximum number of ARP packets you specified, but drops additional packets received within the interval.

To limit the number of ARP packets the device will accept each second, enter the **rate-limit-arp** command at the global CONFIG level of the CLI.

```
device(config)# rate-limit-arp 100
```

This command configures the device to accept up to 100 ARP packets each second. If the device receives more than 100 ARP packets during a one-second interval, the device drops the additional ARP packets during the remainder of that one-second interval.

Syntax:[no] **rate-limit-arp** *num*

The *num* variable specifies the number of ARP packets and can be from 0 through 100. If you specify 0, the device will not accept any ARP packets.

NOTE

If you want to change a previously configured the ARP rate limiting policy, you must remove the previously configured policy using the **no rate-limit-arp** command before entering the new policy.

Changing the ARP aging period

When the Layer 3 switch places an entry in the ARP cache, the Layer 3 switch also starts an aging timer for the entry. 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.

The ARP age affects dynamic (learned) entries only, not static entries. The default ARP age is ten minutes. On Layer 3 switches, you can change the ARP age to a value from 0 through 240 minutes. You cannot change the ARP age on Layer 2 switches. If you set the ARP age to zero, aging is disabled and entries do not age out.

NOTE

Host devices connected to an ICX 7750 that also have a valid IP address and reply periodically to the arp request are not timed out, even if no traffic is destined for the device. This behavior is restricted to only ICX 7750 devices.

To globally change the ARP aging parameter to 20 minutes, enter the **ip arp-age** command.

```
device(config)# ip arp-age 20
```

Syntax: [no] **ip arp-age** *num*

The *num* parameter specifies the number of minutes, which can be from 0 through 240. The default is 10. If you specify 0, aging is disabled.

To override the globally configured IP ARP age on an individual interface, enter the **ip arp-age** command followed by the new value at the interface configuration level.

```
device(config-if-e1000-1/1/1)# ip arp-age 30
```

Enabling proxy ARP

Proxy ARP allows a Layer 3 switch to answer ARP requests from devices on 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. Thus, ARP requests do not cross routers.

For example, if Proxy ARP is enabled on a Layer 3 switch connected to two subnets, 10.10.10.0/24 and 10.20.20.0/24, the Layer 3 switch can respond to an ARP request from 10.10.10.69 for the MAC address of the device with 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 if the subnets are on different network cables, and thus is not answered.

NOTE

An ARP request from one subnet can reach another subnet when both subnets are on the same physical segment (Ethernet cable), because MAC-layer broadcasts reach all the devices on the segment.

Proxy ARP is disabled by default on Brocade Layer 3 switches. This feature is not supported on Brocade Layer 2 switches.

You can enable proxy ARP at the Interface level, as well as at the Global CONFIG level, of the CLI.

NOTE

Configuring proxy ARP at the Interface level overrides the global configuration.

Enabling proxy ARP globally

To enable IP proxy ARP on a global basis, enter the **ip proxy-arp** command.

```
device(config)# ip proxy-arp
```

To again disable IP proxy ARP on a global basis, enter the **no ip proxy-arp** command.

```
device(config)# no ip proxy-arp
```

Syntax: [no] ip proxy-arp

Enabling IP ARP on an interface

NOTE

Configuring proxy ARP at the Interface level overrides the global configuration.

To enable IP proxy ARP on an interface, enter the following commands.

```
device(config)# interface ethernet 5
device(config-if-e1000-5)# ip proxy-arp enable
```

To again disable IP proxy ARP on an interface, enter the following command.

```
device(config)# interface ethernet 5
device(config-if-e1000-5)# ip proxy-arp disable
```

Syntax: [no] ip proxy-arp { enable | disable }

NOTE

By default, gratuitous ARP is disabled for local proxy ARP.

Creating static ARP entries

Static ARP entries are added to the ARP cache when they are configured. Static ARP entries are useful in cases where you want to pre-configure an entry for a device that is not connected to the Layer 3 switch, or you want to prevent a particular entry from aging out.

Brocade Layer 3 switches have a static ARP table, in addition to the regular ARP cache. Unlike static ARP entries, dynamic ARP entries are removed from the ARP cache if the ARP aging interval expires before the entry is refreshed. Static entries do not age out, regardless of whether the Brocade device receives an ARP request from the device that has the entry address.

NOTE

You cannot create static ARP entries on a Layer 2 switch.

The maximum number of static ARP entries you can configure depends on the software version running on the device.

To create a static ARP entry, enter a command such as the following.

```
device(config)# arp 1 10.53.4.2 0000.0054.2348 ethernet 1/1/2
```

Syntax: `arp num ip-addr mac-addr ethernet port`

The *num* variable specifies the entry number. You can specify a number from 1 up to the maximum number of static entries allowed on the device.

The *ip-addr* variable specifies the IP address of the device that has the MAC address of the entry.

The *mac-addr* variable specifies the MAC address of the entry.

Changing the maximum number of entries the static ARP table can hold

NOTE

The basic procedure for changing the static ARP table size is the same as the procedure for changing other configurable cache or table sizes.

To increase the maximum number of static ARP table entries you can configure on a Brocade Layer 3 switch, enter commands such as the following at the global CONFIG level of the CLI.

```
device(config)# system-max ip-static-arp 1000
device(config)# write memory
device(config)# end
device# reload
```

NOTE

You must save the configuration to the startup-config file and reload the software after changing the static ARP table size to place the change into effect.

Syntax: `system-max ip-static-arp num`

The *num* variable indicates the maximum number of static ARP entries and can be within one of these ranges, depending on the software version running on the device.

TABLE 2 Static ARP entry support

Device	Default maximum	Configurable minimum	Configurable maximum
ICX 7250	512	512	6000
ICX 7450	512	512	6000
ICX 7750	512	512	6000

Enabling learning gratuitous ARP

Learning gratuitous ARP enables Brocade Layer 3 devices to learn ARP entries from incoming gratuitous ARP packets from the hosts which are directly connected. This help achieve faster convergence for the hosts when they are ready to send traffic.

A new ARP entry is created when a gratuitous ARP packet is received. If the ARP is already existing, it will be updated with the new content.

To enable learning gratuitous ARP, enter the following command at the device configuration level.

```
Brocade (config)# ip arp learn-gratuitous-arp
```

Syntax: `[no] ip arp learn-gratuitous-arp`

The **no** form of the command disables learning gratuitous ARP from the device.

Use the **show run** command to see whether ARP is enabled or disabled. Use the **show arp** command to see the newly learned ARP entries.

Use the **clear arp** command to clear learned ARP entries. Static ARP entries are not removed.

ARP Packet Validation

You can enable validation options that check incoming ARP packets to avoid traffic interruption or loss.

To avoid traffic interruption or loss, ARP Packet Validation allows the user to detect and drop ARP packets that do not pass the ARP validation process. ARP Packet Validation is disabled by default and can be enabled at the global configuration level. This functionality can be configured for the destination MAC address, the IP address and the source MAC address or with a combination of these parameters. The Ethernet header contains the destination MAC address and source MAC address, while the ARP packet contains the sender hardware address and target hardware address.

Follow these steps to perform checks on incoming ARP packets.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **ip arp inspection validate** command followed by one or more of the available options to perform a check on incoming ARP packets.

- **dst-mac**

The destination MAC address in the Ethernet header must match the target hardware address in the body of ARP response packets. Packets with different MAC addresses are classified as invalid and are dropped.

- **src-mac**

The source MAC address in the Ethernet header must match the sender hardware address in the body of ARP request and response packets. Packets with different MAC addresses are classified as invalid and are dropped.

- **ip**

Each ARP packet has a valid sender IP address and target IP address. In ARP response packets, the target IP address cannot be an invalid or unexpected IP address. The sender IP address cannot be an invalid or unexpected IP address in ARP request or response packets. Addresses include 0.0.0.0, 255.255.255.255, and all IP multicast addresses. Packets with invalid and unexpected IP addresses are classified as invalid and are dropped.

The following example enables validation of ARP packets based on the destination MAC address.

```
device(config)# configuration terminal
device(config)# ip arp inspection validate dst-mac
```

Ingress ARP packet priority

You can configure the priority of the ingress ARP packets to an optimum value that depends on your network configuration and traffic volume. Ingress ARP packets have a default priority value of 4. At the default priority value, ingress ARP packets may get dropped because of high traffic volume or non-ARP packets with higher priority values. This can cause devices to become unreachable. If the ingress ARP packets have higher priority values than the default priority value, a high volume of ARP traffic may lead to drops in control traffic. This may cause traffic loops in the network.

NOTE

You cannot change the priority of the ingress ARP packets on the management port.

Configuring the priority of ingress ARP packets

To configure the priority of ingress ARP packets, use the **arp-internal-priority** *priority-value* command in global configuration mode.

The following example shows the priority of ingress ARP packets set to level 7.

```
Brocade(config)# arp-internal-priority 7
```

Displaying the ARP table

To display the ARP table, enter the **show arp** command.

```
device# show arp
Total number of ARP entries: 2
Entries in default routing instance:
No.   IP Address      MAC Address   Type      Age Port      Status
1     10.1.1.100      0000.0000.0100 Dynamic    0  1/1/1*2/1/25 Valid
2     10.37.69.129    02e0.5215.cae3 Dynamic    0  mgmt1      Valid
```

The command displays all ARP entries in the system.

Syntax: show arp

Reverse Address Resolution Protocol configuration

The Reverse Address Resolution Protocol (RARP) provides a simple mechanism for directly-attached IP hosts to boot over the network. RARP allows an IP host that does not have a means of storing its IP address across power cycles or software reloads to query a directly-attached router for an IP address.

RARP is enabled by default. However, you must create a RARP entry for each host that will use the Layer 3 switch for booting. A RARP entry consists of the following information:

- The entry number - The entry sequence number in the RARP table.
- The MAC address of the boot client.
- The IP address you want the Layer 3 switch to give to the client.

When a client sends a RARP broadcast requesting an IP address, the Layer 3 switch responds to the request by looking in the RARP table for an entry that contains the client MAC address:

- If the RARP table contains an entry for the client, the Layer 3 switch sends a unicast response to the client that contains the IP address associated with the client MAC address in the RARP table.
- If the RARP table does not contain an entry for the client, the Layer 3 switch silently discards the RARP request and does not reply to the client.

How RARP Differs from BootP and DHCP

RARP, BootP, and DHCP are different methods for providing IP addresses to IP hosts when they boot. These methods differ in the following ways:

- Location of configured host addresses
 - RARP requires static configuration of the host IP addresses on the Layer 3 device. The Layer 3 device replies directly to a host request by sending an IP address you have configured in the RARP table.

- The Layer 3 device forwards BootP and DHCP requests to a third-party BootP/DHCP server that contains the IP addresses and other host configuration information.
- Connection of host to boot source (Layer 3 device or BootP/DHCP server)
 - RARP requires the IP host to be directly attached to the Layer 3 device.
 - An IP host and the BootP/DHCP server can be on different networks and on different routers as long as the routers are configured to forward ("help") the host boot request to the boot server.
 - You can centrally configure other host parameters on the BootP/DHCP server and supply those parameters to the host along with its IP address.

To configure the Layer 3 device to forward BootP/DHCP requests when boot clients and boot servers are on different subnets on different Layer 3 device interfaces, refer to the DHCP client section in the *Brocade FastIron Configuration Guide*.

Disabling RARP

RARP is enabled by default. To disable RARP, enter the following command at the global CONFIG level.

```
device(config)# no ip rarp
```

Syntax: [no] ip rarp

To re-enable RARP, enter the following command.

```
device(config)# ip rarp
```

Creating static RARP entries

You must configure the RARP entries for the RARP table. The Layer 3 switch can send an IP address in reply to a client RARP request only if create a RARP entry for that client.

To assign a static IP RARP entry for static routes on a Brocade router, enter a command such as the following.

```
device(config)# rarp 1 0000.0054.2348 10.53.4.2
```

This command creates a RARP entry for a client with MAC address 0000.0054.2348. When the Layer 3 switch receives a RARP request from this client, the Layer 3 switch replies to the request by sending IP address 192.53.4.2 to the client.

Syntax: rap *number mac-addr ip-addr*

The *number* parameter identifies the RARP entry number. You can specify an unused number from 1 to the maximum number of RARP entries supported on the device. To determine the maximum number of entries supported on the device, refer to the section "Displaying and modifying system parameter default settings" in the *Brocade FastIron Platform and Layer 2 Switching Configuration Guide*.

The *mac-addr* parameter specifies the MAC address of the RARP client.

The *ip-addr* parameter specifies the IP address the Layer 3 switch will give the client in response to the client RARP request.

Changing the maximum number of static RARP entries supported

The number of RARP entries the Layer 3 switch supports depends on how much memory the Layer 3 switch has. To determine how many RARP entries your Layer 3 switch can have, display the system default information using the procedure in the section "Displaying system parameter default values" in the *Brocade FastIron Platform and Layer 2 Switching Configuration Guide*.

If your Layer 3 switch allows you to increase the maximum number of RARP entries, you can use a procedure in the same section to do so.

NOTE

You must save the configuration to the startup-config file and reload the software after changing the RARP cache size to place the change into effect.

Dynamic ARP inspection

For enhanced network security, you can configure the Brocade device to inspect and keep track of Dynamic Host Configuration Protocol (DHCP) assignments.

Dynamic ARP Inspection (DAI) enables the Brocade device to intercept and examine all ARP request and response packets in a subnet and discard packets with invalid IP-to-MAC address bindings. DAI can prevent common man-in-the-middle (MiM) attacks such as ARP cache poisoning, and disallow mis-configuration of client IP addresses.

DAI allows only valid ARP requests and responses to be forwarded and supports Multi-VRFs with overlapping address spaces. For more information on DAI, refer to the *Brocade FastIron Security Configuration Guide*.

ARP poisoning

ARP provides IP communication within a Layer 2 broadcast domain by mapping an IP address to a MAC address. Before a host can talk to another host, it must map the IP address to a MAC address first. If the host does not have the mapping in its ARP table, it creates an ARP request to resolve the mapping. All computers on the subnet will receive and process the ARP requests, and the host whose IP address matches the IP address in the request will send an ARP reply.

An ARP poisoning attack can target hosts, switches, and routers connected to the Layer 2 network by poisoning the ARP caches of systems connected to the subnet and by intercepting traffic intended for other hosts on the subnet. For instance, a malicious host can reply to an ARP request with its own MAC address, thereby causing other hosts on the same subnet to store this information in their ARP tables or replace the existing ARP entry. Furthermore, a host can send gratuitous replies without having received any ARP requests. A malicious host can also send out ARP packets claiming to have an IP address that actually belongs to another host (for example, the default router). After the attack, all traffic from the device under attack flows through the attacker computer and then to the router, switch, or host.

Dynamic ARP Inspection

Dynamic ARP Inspection (DAI) allows only valid ARP requests and responses to be forwarded.

A Brocade device on which DAI is configured does the following:

- Intercepts ARP packets received by the system CPU
- Inspects all ARP requests and responses received on untrusted ports
- Verifies that each of the intercepted packets has a valid IP-to-MAC address binding before updating the local ARP table, or before forwarding the packet to the appropriate destination
- Drops invalid ARP packets

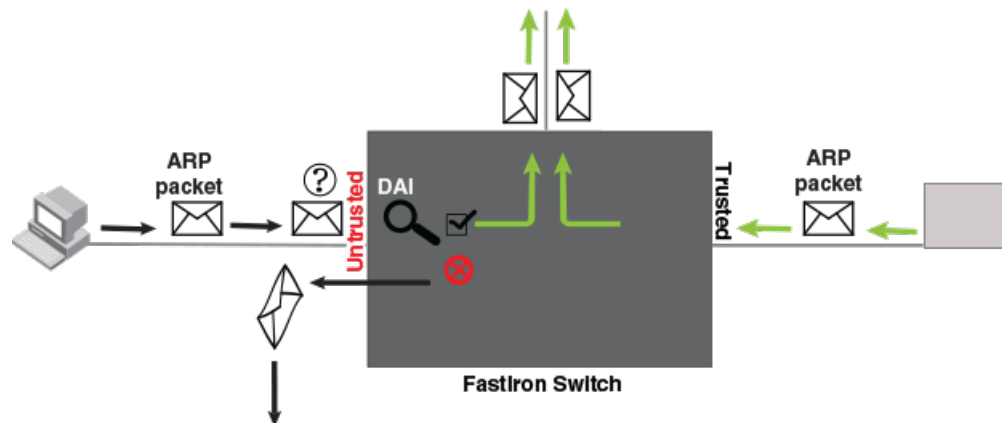
When you enable DAI on a VLAN, by default, all member ports are untrusted. You must manually configure trusted ports. In a typical network configuration, ports connected to host ports are untrusted. You configure ports connected to other switches or routers as trusted.

DAI inspects ARP packets received on untrusted ports, as shown in the figure below. DAI carries out the inspection based on IP-to-MAC address bindings stored in a trusted binding database. For the Brocade device, the binding database is the ARP table and the DHCP snooping table, which supports DAI, DHCP snooping, and IP Source Guard. To inspect an ARP request packet, DAI checks the source IP

address and source MAC address against the ARP table. For an ARP reply packet, DAI checks the source IP, source MAC, destination IP, and destination MAC addresses. DAI forwards the valid packets and discards those with invalid IP-to-MAC address bindings.

When ARP packets reach a trusted port, DAI lets them through, as shown in the following figure.

FIGURE 2 Dynamic ARP inspection at work



ARP and DHCP snoop entries

DAI uses the IP-to-MAC mappings in the ARP table to validate ARP packets received on untrusted ports. DAI relies on the following entries:

- Dynamic ARP - Normal ARP learned from trusted ports.
- Static ARP - Statically configured IP/MAC/port mapping.
- Inspection ARP - Statically configured IP-to-MAC mapping, where the port is initially unspecified. The actual physical port mapping will be resolved and updated from validated ARP packets. Refer to [Configuring an inspection ARP entry](#) on page 33.
- DHCP-Snooping ARP - Information collected from snooping DHCP packets when DHCP snooping is enabled on VLANs. DHCP snooping entries are stored in a different table and are not part of the ARP table.

The status of an ARP entry is either pending or valid:

- Valid - The mapping is valid, and the port is resolved. This is always the case for static ARP entries.
- Pending - For normal dynamic ARP entries before they are resolved, and the port is mapped. Their status changes to valid when they are resolved, and the port is mapped.

Refer to System reboot and the binding database section in the *Brocade FastIron DHCP Configuration Guide*.

Configuration notes and feature limitations for DAI

The following configuration notes and limitations apply when configuring DAI:

- To run Dynamic ARP Inspection, you must first enable support for ACL filtering based on VLAN membership or VE port membership. To do so, enter the following commands at the global configuration level of the CLI.

```
device(config)# enable ACL-per-port-per-vlan
device(config)# write memory
device(config)# exit
device# reload
```


NOTE

You must save the configuration and reload the software to place the change into effect.

- There is a limit on the number of static ARP inspection entries that can be configured. This is determined by the system-max parameter **max-static-inspect-arp-entries**. The maximum value is 1024 and the default value is 512. Changing the system max values requires a system reload.
- ACLs are supported on member ports of a VLAN on which DHCP snooping and Dynamic ARP Inspection (DAI) are enabled.
- DAI is supported on a VLAN without a VE, or on a VE with or without an assigned IP address.
- DAI is supported on LAG ports.

Dynamic ARP Inspection configuration

Configuring DAI consists of the following steps.

1. Configure inspection ARP entries for hosts on untrusted ports. Refer to [Configuring an inspection ARP entry](#) on page 33.
2. Enable DAI on a VLAN to inspect ARP packets. Refer to [Enabling DAI on a VLAN](#) on page 33.
3. Configure the trust settings of the VLAN members. ARP packets received on trusted ports bypass the DAI validation process. ARP packets received on untrusted ports go through the DAI validation process. Refer to [Enabling trust on a port](#) on page 34.
4. Enable DHCP snooping to populate the DHCP snooping IP-to-MAC address binding database.

Dynamic ARP inspection is disabled by default and the trust setting for ports is by default untrusted.

Configuring an inspection ARP entry

Static ARP and static inspection ARP entries must be configured for hosts on untrusted ports. Otherwise, when DAI checks ARP packets from these hosts against entries in the ARP table, it will not find any entries for them, and the Brocade device will not allow and learn ARP from an untrusted host.

To configure an inspection ARP entry, enter a command such as the following.

```
device(config)# arp 10.20.20.12 0000.0002.0003 inspection
```

This command defines an inspection ARP entry in the static ARP table, mapping a device IP address 10.20.20.12 with its MAC address 0000.0002.0003. The ARP entry will be moved to the ARP table once the DAI receives a valid ARP packet.

Dynamic ARP Inspection must be enabled to use static ARP inspection entries.

Syntax: **[no] arp ip-addr mac-addr inspection**

The *ip-addr mac-addr* parameter specifies a device IP address and MAC address pairing.

Enabling DAI on a VLAN

DAI is disabled by default. To enable DAI on an existing VLAN, enter the following command.

```
device(config)# ip arp inspection vlan 2
```

The command enables DAI on VLAN 2. ARP packets from untrusted ports in VLAN 2 will undergo DAI inspection.

Syntax: **[no] ip arp inspection vlan vlan-number**

The *vlan-number* variable specifies the ID of a configured VLAN.

Enabling trust on a port

The default trust setting for a port is untrusted. For ports that are connected to host ports, leave their trust settings as untrusted. If the port is part of a LAG, enable ARP inspection trust on the primary port of the LAG.

To enable trust on a port, enter commands such as the following.

```
device(config)# interface ethernet 1/1/4
device(config-if-e10000-1/1/4)# arp inspection trust
```

The commands change the CLI to the interface configuration level of port 1/1/4 and set the trust setting of port 1/1/4 to trusted.

Syntax: [no] arp inspection trust

Disabling or re-enabling syslog messages for DAI

You can disable or re-enable syslog messages for Dynamic ARP Inspection. Syslog messages are enabled by default on the device.

1. Enter global configuration mode.
2. Enter the **ip arp inspection syslog disable** command to disable syslog messages. Use the **no** form of the command to re-enable syslog messages for DAI.

The following example shows disabling the syslog messages for DAI.

```
device(config)# ip arp inspection syslog disable
```

Multi-VRF support for DAI

DAI supports Multi-VRF (Virtual Routing and Forwarding) instances. You can deploy multiple VRFs on a Brocade Ethernet switch. Each VLAN having a Virtual Ethernet (VE) interface is assigned to a VRF.

You can enable DAI on individual VLANs and assign any interface as the ARP inspection trust interface. If an interface is a tagged port in this VLAN, you can turn on the trust port per VRF, so that traffic intended for other VRF VLANs will not be trusted.

To configure DAI to support a VRF instance, do the following:

- Enable the **acl-per-port-per-vlan** setting. DAI requires that the **acl-per-port-per-vlan** setting be enabled.

```
Brocade(config)# enable acl-per-port-per-vlan
Reload required. Please write memory and then reload or power cycle.
```

- Configure DAI on a VLAN using the **ip arp inspection vlan *vlan-id*** command.

```
Brocade(config)# ip arp inspection vlan 2
```

Syntax: ip arp inspection vlan *vlan-id*

- To add a static ARP inspection entry for a specific VRF, use the **arp *ip-address mac-address* inspection** command in the VRF CLI context.

```
Brocade(config-vrf-one-ipv4)# arp 5.5.5.5 00a2.bbba.0033 inspection
```

Syntax: arp *ip-address mac-address* inspection

Enabling trust on a port for a specific VRF

The default trust setting for a port is untrusted. For ports that are connected to host ports, leave their trust settings as untrusted.

To enable trust on a port for a specific VRF, enter commands such as the following.

```
Brocade(config)# interface ethernet 1/1/4
Brocade(config-if-e10000-1/1/4)# arp inspection trust vrf vrf2
```

The commands change the CLI to the interface configuration level of port 1/1/4 and set the trust setting of port 1/1/4 on VRF 2 to trusted.

Syntax: `[no] arp inspection trust vrf vrf-name`

Displaying ARP inspection status and ports

To display the ARP inspection status for a VLAN and the trusted or untrusted port, enter the following command.

```
device# show ip arp inspection vlan 2
IP ARP inspection VLAN 2: Disabled
Trusted Ports :   ethe 1/1/4
Untrusted Ports : ethe 2/1/1 to 2/1/3 ethe 4/1/1 to 4/1/24 ethe 6/1/1 to 6/1/4 ethe 8/1/1 to 8/1/4
```

Syntax: `show ip arp inspection vlan vlan_id`

The *vlan_id* variable specifies the ID of a configured VLAN.

IP Addressing

• IP configuration overview.....	37
• Basic IP parameters and defaults - Layer 3 switches.....	44
• Basic IP parameters and defaults - Layer 2 switches.....	49
• Basic IP configuration.....	51
• IP address replacement.....	51
• Configuring IP parameters - Layer 3 switches.....	52
• Configuring IP parameters - Layer 2 switches.....	82
• IPv4 point-to-point GRE tunnels	85
• Bandwidth for IP interfaces.....	100
• User-configurable MAC address per IP interface.....	103
• Modifying and displaying Layer 3 system parameter limits.....	105
• Enabling or disabling routing protocols.....	106
• Enabling or disabling Layer 2 switching.....	107
• Configuring a Layer 3 Link Aggregation Group (LAG).....	107
• Disabling IP checksum check.....	108
• Displaying IP configuration information and statistics.....	109

IP configuration overview

IPv4 uses a 32-bit addressing system designed for use in packet-switched networks. IPv4 is the Internet protocol that is most commonly used currently throughout the world. IPv4 uses a 32-bit addressing system and is represented in a 4-byte dotted decimal format: x.x.x.x.

Brocade Layer 2 switches and Layer 3 switches support Internet Protocol version 4 (IPv4) and IPv6. IP support on Brocade Layer 2 switches consists of basic services to support management access and access to a default gateway.

Full Layer 3 support

IP support on Brocade full Layer 3 switches includes all of the following, in addition to a highly configurable implementation of basic IP services including Address Resolution Protocol (ARP), ICMP Router Discovery Protocol (IRDP), and Reverse ARP (RARP):

- Route exchange protocols:
 - Routing Information Protocol (RIP)
 - Open Shortest Path First (OSPF)
 - Border Gateway Protocol version 4 (BGP4)
- Multicast protocols:
 - Internet Group Management Protocol (IGMP)
 - Protocol Independent Multicast Dense (PIM-DM)
 - Protocol Independent Multicast Sparse (PIM-SM)
- Router redundancy protocols:
 - Virtual Router Redundancy Protocol Extended (VRRP-E)
 - Virtual Router Redundancy Protocol (VRRP)

IP interfaces

NOTE

This section describes IPv4 addresses. For information about IPv6 addresses, refer to the *IPv6 addressing* chapter.

Brocade Layer 3 switches and Layer 2 switches allow you to configure IP addresses. On Layer 3 switches, IP addresses are associated with individual interfaces. On Layer 2 switches, a single IP address serves as the management access address for the entire device.

All Brocade Layer 3 switches and Layer 2 switches support configuration and display of IP addresses in classical subnet format (for example, 192.168.1.1 255.255.255.0) and Classless Interdomain Routing (CIDR) format (for example, 192.168.1.1/24). You can use either format when configuring IP address information. IP addresses are displayed in classical subnet format by default but you can change the display format to CIDR.

Layer 3 switches

Brocade Layer 3 switches allow you to configure IP addresses on the following types of interfaces:

- Ethernet ports
- Virtual routing interfaces (used by VLANs to route among one another)
- Loopback interfaces
- GRE tunnels

Each IP address on a Layer 3 switch must be in a different subnet. You can have only one interface that is in a given subnet. For example, you can configure IP addresses 192.168.1.1/24 and 192.168.2.1/24 on the same Layer 3 switch, but you cannot configure 192.168.1.1/24 and 192.168.1.2/24 on the same Layer 3 switch.

You can configure multiple IP addresses on the same interface.

The number of IP addresses you can configure on an individual interface depends on the Layer 3 switch model. To display the maximum number of IP addresses and other system parameters you can configure on a Layer 3 switch, refer to "Displaying and modifying system parameter default settings" section in the *Brocade FastIron Platform and Layer 2 Switching Configuration Guide*.

You can use any of the IP addresses you configure on the Layer 3 switch for Telnet, Web management, or SNMP access.

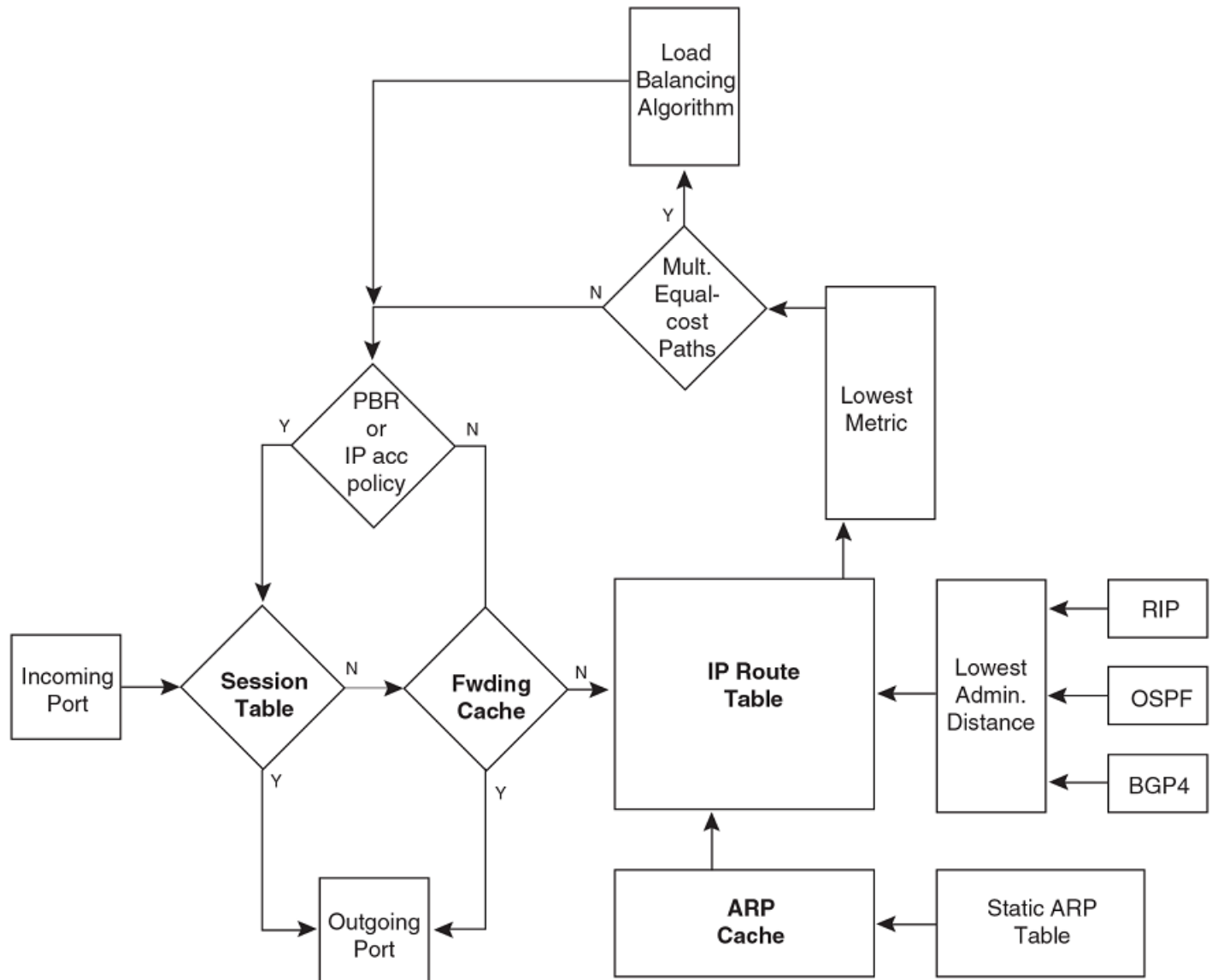
Layer 2 switches

You can configure an IP address on a Brocade Layer 2 switch for management access to the Layer 2 switch. An IP address is required for Telnet access, Web management access, and SNMP access.

You also can specify the default gateway for forwarding traffic to other subnets.

IP packet flow through a Layer 3 switch

FIGURE 3 IP Packet flow through a Brocade Layer 3 switch



1. When the Layer 3 switch receives an IP packet, the Layer 3 switch checks for filters on the receiving interface.¹ If a deny filter on the interface denies the packet, the Layer 3 switch discards the packet and performs no further processing, except generating a Syslog entry and SNMP message, if logging is enabled for the filter.

¹ The filter can be an Access Control List (ACL) or an IP access policy.

2. If the packet is not denied at the incoming interface, the Layer 3 switch looks in the session table for an entry that has the same source IP address and TCP or UDP port as the packet. If the session table contains a matching entry, the Layer 3 switch immediately forwards the packet, by addressing it to the destination IP address and TCP or UDP port listed in the session table entry and sending the packet to a queue on the outgoing ports listed in the session table. The Layer 3 switch selects the queue based on the Quality of Service (QoS) level associated with the session table entry.
3. If the session table does not contain an entry that matches the packet source address and TCP or UDP port, the Layer 3 switch looks in the IP forwarding cache for an entry that matches the packet destination IP address. If the forwarding cache contains a matching entry, the Layer 3 switch forwards the packet to the IP address in the entry. The Layer 3 switch sends the packet to a queue on the outgoing ports listed in the forwarding cache. The Layer 3 switch selects the queue based on the Quality of Service (QoS) level associated with the forwarding cache entry.
4. If the IP forwarding cache does not have an entry for the packet, the Layer 3 switch checks the IP route table for a route to the packet destination. If the IP route table has a route, the Layer 3 switch makes an entry in the session table or the forwarding cache, and sends the route to a queue on the outgoing ports:
 - If the running-config contains an IP access policy for the packet, the software makes an entry in the session table. The Layer 3 switch uses the new session table entry to forward subsequent packets from the same source to the same destination.
 - If the running-config does not contain an IP access policy for the packet, the software creates a new entry in the forwarding cache. The Layer 3 switch uses the new cache entry to forward subsequent packets to the same destination.

The following sections describe the IP tables and caches:

- ARP cache and static ARP table
- IP route table
- IP forwarding cache
- Layer 4 session table

The software enables you to display these tables. You also can change the capacity of the tables on an individual basis if needed by changing the memory allocation for the table.

ARP cache and static ARP table

The ARP cache contains entries that map IP addresses to MAC addresses. Generally, the entries are for devices that are directly attached to the Layer 3 switch.

An exception is an ARP entry for an interface-based static IP route that goes to a destination that is one or more router hops away. For this type of entry, the MAC address is either the destination device MAC address or the MAC address of the router interface that answered an ARP request on behalf of the device, using proxy ARP.

ARP cache

The ARP cache can contain dynamic (learned) entries and static (user-configured) entries. The software places a dynamic entry in the ARP cache when the Layer 3 switch learns a device MAC address from an ARP request or ARP reply from the device.

The software can learn an entry when the Layer 2 switch or Layer 3 switch receives an ARP request from another IP forwarding device or an ARP reply. Here is an example of a dynamic entry:

IP Address	MAC Address	Type	Age	Port	
1	10.95.6.102	0000.00fc.ea21	Dynamic	0	6

Each entry contains the destination device IP address and MAC address.

Static ARP table

In addition to the ARP cache, Layer 3 switches have a static ARP table. Entries in the static ARP table are user-configured. You can add entries to the static ARP table regardless of whether or not the device the entry is for is connected to the Layer 3 switch.

NOTE

Layer 3 switches have a static ARP table. Layer 2 switches do not.

The software places an entry from the static ARP table into the ARP cache when the entry interface comes up.

Here is an example of a static ARP entry.

Index	IP Address	MAC Address	Port
1	10.95.6.111	0000.003b.d210	1/1/1

Each entry lists the information you specified when you created the entry.

IP route table

The IP route table contains paths to IP destinations.

NOTE

Layer 2 switches do not have an IP route table. A Layer 2 switch sends all packets addressed to another subnet to the default gateway, which you specify when you configure the basic IP information on the Layer 2 switch.

The IP route table can receive the paths from the following sources:

- A directly-connected destination, which means there are no router hops to the destination
- A static IP route, which is a user-configured route
- A route learned through RIP
- A route learned through OSPF
- A route learned through BGP4

The IP route table contains the best path to a destination:

- When the software receives paths from more than one of the sources listed above, the software compares the administrative distance of each path and selects the path with the lowest administrative distance. The administrative distance is a protocol-independent value from 1 through 255.
- When the software receives two or more best paths from the same source and the paths have the same metric (cost), the software can load share traffic among the paths based on destination host or network address (based on the configuration and the Layer 3 switch model).

Here is an example of an entry in the IP route table.

Destination	NetMask	Gateway	Port	Cost	Type
10.1.0.0	255.255.0.0	10.1.1.2	1/1/1	2	R

Each IP route table entry contains the destination IP address and subnet mask and the IP address of the next-hop router interface to the destination. Each entry also indicates the port attached to the destination or the next-hop to the destination, the route IP metric (cost), and the type. The type indicates how the IP route table received the route.

To increase the size of the IP route table for learned and static routes, refer to the section "Displaying and modifying system parameter default settings" in the *Brocade FastIron Layer 2 Switching Configuration Guide*

- For learned routes, modify the *ip-route* parameter.
- For static routes, modify the *ip-static-route* parameter.

IP forwarding cache

The IP forwarding cache provides a fast-path mechanism for forwarding IP packets. The cache contains entries for IP destinations. When a Brocade Layer 3 switch has completed processing and addressing for a packet and is ready to forward the packet, the device checks the IP forwarding cache for an entry to the packet destination:

- If the cache contains an entry with the destination IP address, the device uses the information in the entry to forward the packet out the ports listed in the entry. The destination IP address is the address of the packet final destination. The port numbers are the ports through which the destination can be reached.
- If the cache does not contain an entry and the traffic does not qualify for an entry in the session table instead, the software can create an entry in the forwarding cache.

Each entry in the IP forwarding cache has an age timer. If the entry remains unused for ten minutes, the software removes the entry. The age timer is not configurable.

Here is an example of an entry in the IP forwarding cache.

IP Address	Next Hop	MAC	Type	Port	Vlan	Pri
1 192.168.1.11	DIRECT	0000.0000.0000	PU	n/a		0

Each IP forwarding cache entry contains the IP address of the destination, and the IP address and MAC address of the next-hop router interface to the destination. If the destination is actually an interface configured on the Layer 3 switch itself, as shown here, then next-hop information indicates this. The port through which the destination is reached is also listed, as well as the VLAN and Layer 4 QoS priority associated with the destination if applicable.

NOTE

You cannot add static entries to the IP forwarding cache, although you can increase the number of entries the cache can contain. Refer to the section "Displaying and modifying system parameter default settings" in the *Brocade FastIron Layer 2 Switching Configuration Guide*.

Layer 4 session table

The Layer 4 session provides a fast path for forwarding packets. A session is an entry that contains complete Layer 3 and Layer 4 information for a flow of traffic. Layer 3 information includes the source and destination IP addresses. Layer 4 information includes the source and destination TCP and UDP ports. For comparison, the IP forwarding cache contains the Layer 3 destination address but does not contain the other source and destination address information of a Layer 4 session table entry.

The Layer 2 switch or Layer 3 switch selects the session table instead of the IP forwarding table for fast-path forwarding for the following features:

- Layer 4 Quality-of-Service (QoS) policies
- IP access policies

To increase the size of the session table, refer to the section "Displaying and modifying system parameter default settings" in the *Brocade FastIron Layer 2 Switching Configuration Guide*. The `ip-qos-session` parameter controls the size of the session table.

IP route exchange protocols

Brocade Layer 3 switches support the following IP route exchange protocols:

- Routing Information Protocol (RIP)
- Open Shortest Path First (OSPF)
- Border Gateway Protocol version 4 (BGP4)

All these protocols provide routes to the IP route table. You can use one or more of these protocols, in any combination. The protocols are disabled by default.

IP multicast protocols

Brocade Layer 3 switches also support the following Internet Group Membership Protocol (IGMP) based IP multicast protocols:

- Protocol Independent Multicast - Dense mode (PIM-DM)
- Protocol Independent Multicast - Sparse mode (PIM-SM)

For configuration information, refer to "IP Multicast Protocols" in the *Brocade FastIron IP Multicast Configuration Guide*.

NOTE

Brocade Layer 3 switches support IGMP and can forward IP multicast packets. Refer to the "IP Multicast Traffic Reduction" chapter in the *Brocade FastIron IP Multicast Configuration Guide*.

IP interface redundancy protocols

You can configure a Brocade Layer 3 switch to back up an IP interface configured on another Brocade Layer 3 switch. If the link for the backed up interface becomes unavailable, the other Layer 3 switch can continue service for the interface. This feature is especially useful for providing a backup to a network default gateway.

Brocade Layer 3 switches support the following IP interface redundancy protocols:

- Virtual Router Redundancy Protocol (VRRP) - A standard router redundancy protocol based on RFC 2338. You can use VRRP to configure Brocade Layer 3 switches and third-party routers to back up IP interfaces on other Brocade Layer 3 switches or third-party routers.
- Virtual Router Redundancy Protocol Extended (VRRP-E) - A Brocade extension to standard VRRP that adds additional features and overcomes limitations in standard VRRP. You can use VRRP-E only on Brocade Layer 3 switches.

ACLs and IP access policies

Brocade Layer 3 switches provide two mechanisms for filtering IP traffic:

- Access Control Lists (ACLs)
- IP access policies

Both methods allow you to filter packets based on Layer 3 and Layer 4 source and destination information.

ACLs also provide great flexibility by providing the input to various other filtering mechanisms such as route maps, which are used by BGP4.

IP access policies allow you to configure QoS based on sessions (Layer 4 traffic flows).

Only one of these filtering mechanisms can be enabled on a Brocade device at a time. Brocade devices can store forwarding information for both methods of filtering in the session table.

For configuration information, refer to the chapter "Rule-Based IP ACLs" in the *Brocade FastIron Security Configuration Guide*.

Basic IP parameters and defaults - Layer 3 switches

IP is enabled by default. The following IP-based protocols are all disabled by default:

- Routing protocols:
 - Routing Information Protocol (RIP)
 - Open Shortest Path First (OSPF)
 - Border Gateway Protocol version 4 (BGP4)
- Multicast protocols:
 - Internet Group Membership Protocol (IGMP)
 - Protocol Independent Multicast Dense (PIM-DM)
 - Protocol Independent Multicast Sparse (PIM-SM)
- Router redundancy protocols:
 - Virtual Router Redundancy Protocol Extended (VRRP-E)
 - Virtual Router Redundancy Protocol (VRRP)

When parameter changes take effect

Most IP parameters described in this chapter are dynamic. They take effect immediately, as soon as you enter the CLI command or select the Web Management Interface option. You can verify that a dynamic change has taken effect by displaying the running-config. To display the running-config, enter the **show running-config** or **write terminal** command at any CLI prompt. (You cannot display the running-config from the Web Management Interface.)

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

- To save configuration changes to the startup-config file, enter the **write memory** command from the Privileged EXEC level of any configuration level of the CLI.
- To save the configuration changes using the Web Management Interface, select the Save link at the bottom of the dialog. Select Yes when prompted to save the configuration change to the startup-config file on the device flash memory. You also can access the dialog for saving configuration changes by clicking on Command in the tree view, then clicking on Save to Flash.

Changes to memory allocation require you to reload the software after you save the changes to the startup-config file. When reloading the software is required to complete a configuration change described in this chapter, the procedure that describes the configuration change includes a step for reloading the software.

IP global parameters - Layer 3 switches

TABLE 3 IP global parameters - Layer 3 switches

Parameter	Description	Default
IP state	The Internet Protocol, version 4	Enabled NOTE You cannot disable IP.
IP address and mask notation	Format for displaying an IP address and its network mask information. You can enable one of the following: <ul style="list-style-type: none"> • Class-based format; example: 192.168.1.1 255.255.255.0 	Class-based

TABLE 3 IP global parameters - Layer 3 switches (continued)

Parameter	Description	Default
	<ul style="list-style-type: none"> Classless Interdomain Routing (CIDR) format; example: 192.168.1.1/24 	NOTE Changing this parameter affects the display of IP addresses, but you can enter addresses in either format regardless of the display setting.
Router ID	The value that routers use to identify themselves to other routers when exchanging route information. OSPF and BGP4 use router IDs to identify routers. RIP does not use the router ID.	The IP address configured on the lowest-numbered loopback interface. If no loopback interface is configured, then the lowest-numbered IP address configured on the device.
Maximum Transmission Unit (MTU)	The maximum length an Ethernet packet can be without being fragmented.	1500 bytes for Ethernet II encapsulation 1492 bytes for SNAP encapsulation
Address Resolution Protocol (ARP)	A standard IP mechanism that routers use to learn the Media Access Control (MAC) address of a device on the network. The router sends the IP address of a device in the ARP request and receives the device MAC address in an ARP reply.	Enabled
ARP rate limiting	You can 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 ARP cache. The device resets the timer to zero each time the ARP entry is refreshed and removes the entry if the timer reaches the ARP age. NOTE You also can change the ARP age on an individual interface basis.	10 minutes
Proxy ARP	An IP mechanism a router can use to answer an ARP request on behalf of a host by replying with the router's own MAC address instead of the host.	Disabled
Static ARP entries	An ARP entry you place in the static ARP table. Static entries do not age out.	No entries
Time to Live (TTL)	The maximum number of routers (hops) through which a packet can pass before being discarded. Each router decreases a packet 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.	Disabled

TABLE 3 IP global parameters - Layer 3 switches (continued)

Parameter	Description	Default
	NOTE You also can enable or disable this parameter on an individual interface basis.	
Directed broadcast mode	The packet format the router treats as a directed broadcast. The following formats can be directed broadcasts: <ul style="list-style-type: none"> • All ones in the host portion of the packet destination address. • All zeroes in the host portion of the packet destination address. 	All ones NOTE If you enable all-zeroes directed broadcasts, all-ones directed broadcasts remain enabled.
Source-routed packet forwarding	A source-routed packet contains a list of IP addresses through which the packet must pass to reach its destination.	Enabled
Internet Control Message Protocol (ICMP) messages	The Brocade Layer 3 switch can send the following types of ICMP messages: <ul style="list-style-type: none"> • Echo messages (ping messages) • Destination Unreachable messages 	Enabled
ICMP Router Discovery Protocol (IRDP)	An IP protocol a router can use to advertise the IP addresses of its router interfaces to directly attached hosts. You can enable or disable the protocol, and change the following protocol parameters: <ul style="list-style-type: none"> • Forwarding method (broadcast or multicast) • Hold time • Maximum advertisement interval • Minimum advertisement interval • Router preference level NOTE You also can enable or disable IRDP and configure the parameters on an individual interface basis.	Disabled
Reverse ARP (RARP)	An IP mechanism a host can use to request an IP address from a directly attached router when the host boots.	Enabled
Static RARP entries	An IP address you place in the RARP table for RARP requests from hosts. NOTE You must enter the RARP entries manually. The Layer 3 switch does not have a mechanism for learning or dynamically generating RARP entries.	No entries
Maximum BootP relay hops	The maximum number of hops away a BootP server can be located from a router and still be used by the router clients for network booting.	Four
Domain name for Domain Name Server (DNS) resolver	A domain name (for example, brocade.router.com) you can use in place of an	None configured

TABLE 3 IP global parameters - Layer 3 switches (continued)

Parameter	Description	Default
	IP address for certain operations such as IP pings, trace routes, and Telnet management connections to the router.	
DNS default gateway addresses	A list of gateways attached to the router through which clients attached to the router can reach DNSs.	None configured
IP load sharing	<p>A Brocade feature that enables the router to balance traffic to a specific destination across multiple equal-cost paths.</p> <p>IP load sharing uses a hashing algorithm based on the source IP address, destination IP address, protocol field in the IP header, TCP, and UDP information.</p> <p>You can specify the number of load-sharing paths depending on the device you are configuring. The maximum number of paths the device supports is a value from 2 through 8. The default value is 4. On the Brocade ICX 7750, the value range for the maximum number of load-sharing paths is from 2 through 32 which is controlled by the system-max max-ecmp command.</p> <p>NOTE Load sharing is sometimes called equal-cost multi-path (ECMP).</p>	Enabled
Maximum IP load sharing paths	The maximum number of equal-cost paths across which the Layer 3 switch is allowed to distribute traffic.	Four
Origination of default routes	<p>You can enable a router to originate default routes for the following route exchange protocols, on an individual protocol basis:</p> <ul style="list-style-type: none"> • OSPF • BGP4 	Disabled
Default network route	The router uses the default network route if the IP route table does not contain a route to the destination and also does not contain an explicit default route (0.0.0.0 0.0.0.0 or 0.0.0.0/0).	None configured
Static route	An IP route you place in the IP route table.	No entries
Source interface	<p>The IP address the router uses as the source address for Telnet, RADIUS, or TACACS+ packets originated by the router. The router can select the source address based on either of the following:</p> <ul style="list-style-type: none"> • 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 - Layer 3 switches

TABLE 4 IP interface parameters - Layer 3 switches

Parameter	Description	Default
IP state	The Internet Protocol, version 4	Enabled NOTE You cannot disable IP.
IP address	A Layer 3 network interface address NOTE Layer 2 switches have a single IP address used for management access to the entire device. Layer 3 switches have separate IP addresses on individual interfaces.	None configured NOTE Some devices have a factory default, such as 10.157.22.154, used for troubleshooting during installation. For Layer 3 switches, the address is on unit 1/slot 1/ port 1 (or 1/1/1).
Encapsulation type	The format of the packets in which the router encapsulates IP datagrams. The encapsulation format can be one of the following: <ul style="list-style-type: none"> Ethernet II SNAP 	Ethernet II
Maximum Transmission Unit (MTU)	The maximum length (number of bytes) of an encapsulated IP datagram the router can forward.	1500 for Ethernet II encapsulated packets 1492 for SNAP encapsulated packets
Delay L3 notifications	When all ports in the VLAN go into the non-forwarding state, the device waits for the configured time before notifying the Layer 3 protocols of the VE down event. NOTE Available on the VE interface only.	Delay time is not configured
ARP age	Locally overrides the global setting.	Ten minutes
Directed broadcast forwarding	Locally overrides the global setting.	Disabled
ICMP Router Discovery Protocol (IRDP)	Locally overrides the global IRDP settings.	Disabled
DHCP gateway stamp	The router can assist DHCP/BootP Discovery packets from one subnet to reach DHCP/BootP servers on a different subnet by placing the IP address of the router interface that receives the request in the request packet Gateway field. You can override the default and specify the IP address to use for the Gateway field in the packets. NOTE UDP broadcast forwarding for client DHCP/BootP requests (bootps) must be enabled (this is enabled by default) and you must configure an IP helper address (the server IP address or a directed broadcast to the server subnet) on the port connected to the client.	The lowest-numbered IP address on the interface that receives the request

TABLE 4 IP interface parameters - Layer 3 switches (continued)

Parameter	Description	Default
DHCP Client-Based Auto-Configuration	Allows the switch to obtain IP addresses from a DHCP host automatically, for either a specified (leased) or infinite period of time.	Enabled
DHCP Server	All FastIron devices can be configured to function as DHCP servers.	Disabled
UDP broadcast forwarding	<p>The router can forward UDP broadcast packets for UDP applications such as BootP. By forwarding the UDP broadcasts, the router enables clients on one subnet to find servers attached to other subnets.</p> <p>NOTE To completely enable a client UDP application request to find a server on another subnet, you must configure an IP helper address consisting of the server IP address or the directed broadcast address for the subnet that contains the server. Refer to the next row.</p>	<p>The router helps forward broadcasts for the following UDP application protocols:</p> <ul style="list-style-type: none"> • 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 router to forward requests for certain UDP applications from a client on one subnet to a server on another subnet.	None configured

Basic IP parameters and defaults - Layer 2 switches

IP is enabled by default. The following tables list the Layer 2 switch IP parameters, their default values, and where to find configuration information.

NOTE

Brocade Layer 2 switches also provide IP multicast forwarding, which is enabled by default. For information about this feature, refer to "IP Multicast Traffic Reduction" in the *Brocade FastIron IP Multicast Configuration Guide*.

IP global parameters - Layer 2 switches

TABLE 5 IP global parameters - Layer 2 switches

Parameter	Description	Default
IP address and mask notation	<p>Format for displaying an IP address and its network mask information. You can enable one of the following:</p> <ul style="list-style-type: none"> • Class-based format; example: 192.168.1.1 255.255.255.0 • Classless Interdomain Routing (CIDR) format; example: 192.168.1.1/24 	<p>Class-based</p> <p>NOTE Changing this parameter affects the display of IP addresses, but you can enter addresses in either format regardless of the display setting.</p>
IP address	A Layer 3 network interface address	None configured

TABLE 5 IP global parameters - Layer 2 switches (continued)

Parameter	Description	Default
	NOTE Layer 2 switches have a single IP address used for management access to the entire device. Layer 3 switches have separate IP addresses on individual interfaces.	NOTE Some devices have a factory default, such as 10.157.22.154, used for troubleshooting during installation. For Layer 3 switches, the address is on port 1 (or 1/1/1).
Default gateway	The IP address of a locally attached router (or a router attached to the Layer 2 switch by bridges or other Layer 2 switches). The Layer 2 switch and clients attached to it use the default gateway to communicate with devices on other subnets.	None configured
Address Resolution Protocol (ARP)	A standard IP mechanism that networking devices use to learn the Media Access Control (MAC) address of another device on the network. The Layer 2 switch sends the IP address of a device in the ARP request and receives the device MAC address in an ARP reply.	Enabled NOTE You cannot disable ARP.
ARP age	The amount of time the device keeps a MAC address learned through ARP in the device 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.	Ten minutes NOTE You cannot change the ARP age on Layer 2 switches.
Time to Live (TTL)	The maximum number of routers (hops) through which a packet can pass before being discarded. Each router decreases a packet 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
Domain name for Domain Name Server (DNS) resolver	A domain name (example: brocade.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 router.	None configured
DNS default gateway addresses	A list of gateways attached to the router through which clients attached to the router can reach DNSs.	None configured
Source interface	The IP address the Layer 2 switch uses as the source address for Telnet, RADIUS, or TACACS+ packets originated by the router. The Layer 2 switch uses its management IP address as the source address for these packets.	The management IP address of the Layer 2 switch. NOTE This parameter is not configurable on Layer 2 switches.
DHCP gateway stamp	The device can assist DHCP/BootP Discovery packets from one subnet to reach DHCP/BootP servers on a different subnet by placing the IP address of the router interface that forwards the packet in the packet Gateway field. You can specify up to 32 gateway lists. A gateway list contains up to eight gateway IP addresses. You activate DHCP assistance by associating a gateway list with a port. When you configure multiple IP addresses in a gateway list, the Layer 2 switch inserts the	None configured

TABLE 5 IP global parameters - Layer 2 switches (continued)

Parameter	Description	Default
	addresses into the DHCP Discovery packets in a round robin fashion.	
DHCP Client-Based Auto-Configuration	Allows the switch to obtain IP addresses from a DHCP host automatically, for either a specified (leased) or infinite period of time.	Enabled

Interface IP parameters - Layer 2 switches

TABLE 6 Interface IP parameters - Layer 2 switches

Parameter	Description	Default
DHCP gateway stamp	You can configure a list of DHCP stamp addresses for a port. When the port receives a DHCP/BootP Discovery packet from a client, the port places the IP addresses in the gateway list into the packet Gateway field.	None configured

Basic IP configuration

IP is enabled by default. Basic configuration consists of adding IP addresses for Layer 3 switches, enabling a route exchange protocol, such as the Routing Information Protocol (RIP).

NOTE

The terms Layer 3 switch and router are used interchangeably in this chapter and mean the same.

If you are configuring a Layer 3 switch, refer to Configuring IP addresses to add IP addresses, then enable and configure the route exchange protocols, as described in other chapters of this guide.

If you are configuring a Layer 2 switch, refer to Configuring the management IP address and specifying the default gateway to add an IP address for management access through the network and to specify the default gateway.

The rest of this chapter describes IP and how to configure it in more detail. Use the information in this chapter if you need to change some of the IP parameters from their default values or you want to view configuration information or statistics.

IP address replacement

An interface supports multiple primary address configuration. However, you can configure only one primary IP address in a subnet. Beginning with Brocade FastIron release 08.0.50, you can configure the primary IP address on a physical interface (Ethernet), management interface, virtual interface (VE or loopback), or a tunnel interface even if the primary IP address was already configured.

When you configure an IP address, using the **ip address** command with the **replace** option, the existing IP address is removed and the new IP address is applied to the interface. After the IP address replacement, you must re-establish all the Telnet and Secure Shell (SSH) sessions because the current sessions get either terminated or timed out.

Limitations and restrictions

The IP address replacement feature has the following limitations and restrictions.

- The IP address replacement feature is not supported for an IPv6 address configuration.
- The IP address replacement feature is supported on a Brocade FastIron router image only.
- You cannot change the subnet mask by using the **replace** parameter.
- You can replace IP addresses only on the management interface and data ports.
- You cannot replace a secondary IP address. You can only replace the primary IP address.

Replacing an IP address

You must remove all secondary IP addresses, if they exist, before replacing the existing primary IP address with a new IP address.

1. NOTE

The device does not prompt, if no primary IP address matching subnet of the new IP address is already configured on interface and the user use the **replace** option. When the **ip address** command is used with the **replace** option, the new IP address is configured.

Enter global configuration mode.

```
device# configure terminal
```

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

```
device (config)# interface ethernet 1/1/21
```

3. Use the **ip address** command to replace an existing primary IP address with a new IP address. Note that when you choose the replace option to remove the existing IP address on an interface, the action cannot be undone.

```
device(config-if-e1000-1/1/21)# ip address 10.1.1.1/24 replace
```

4. Use the **show running interface** command to display the newly configured IP address on the interface.

```
device(config-if-e1000-1/1/21)# show running interface ethernet 1/1/21
```

The **replace** option is not displayed in running or startup configurations.

The following example shows how to replace the primary IP address of an interface.

```
device# configure terminal
device(config)# interface ethernet 1/1/21
device(config-if-e1000-1/1/21)# ip address 10.1.1.1/24 replace
```

The following example shows how to display the primary IP address of an interface.

```
device(config-if-e1000-1/1/21)# show running interface ethernet 1/1/21
interface ethernet 1/1/21
ip address 11.1.1.2 255.255.255.0!
```

Configuring IP parameters - Layer 3 switches

The following sections describe how to configure IP parameters. Some parameters can be configured globally while others can be configured on individual interfaces. Some parameters can be configured globally and overridden for individual interfaces.

Configuring IP addresses

You can configure an IP address on the following types of Layer 3 switch interfaces:

- Ethernet port
- Virtual routing interface (also called a Virtual Ethernet or "VE")
- Loopback interface
- GRE tunnels

By default, you can configure up to 24 IP addresses on each interface.

NOTE

- When you configure an IPv4 address on a device, a syslog appears stating that the IP address has been added.
- If you reconfigure the same IP address or a different IP address on the device, a syslog appears stating that the IP address has been added.
- The syslog does not state that the existing IP address was replaced with a new IP address.

You can increase this amount to up to 128 IP subnet addresses per port by increasing the size of the ip-subnet-port table.

Refer to the section "Displaying system parameter default values" in the *Brocade FastIron Platform and Layer 2 Switching Configuration Guide*.

NOTE

Once you configure a virtual routing interface on a VLAN, you cannot configure Layer 3 interface parameters on individual ports. Instead, you must configure the parameters on the virtual routing interface itself.

Brocade devices support both classical IP network masks (Class A, B, and C subnet masks, and so on) and Classless Interdomain Routing (CIDR) network prefix masks:

- To enter a classical network mask, enter the mask in IP address format. For example, enter "10.157.22.99 255.255.255.0" for an IP address with a Class-C subnet mask.
- To enter a prefix network mask, enter a forward slash (/) and the number of bits in the mask immediately after the IP address. For example, enter "10.157.22.99/24" for an IP address that has a network mask with 24 significant bits (ones).

By default, the CLI displays network masks in classical IP address format (for example, 255.255.255.0). You can change the display to prefix format.

Assigning an IP address to an Ethernet port

To assign an IP address to port 1/1/1, enter the following commands.

```
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1)# ip address 10.45.6.1 255.255.255.0
```

You also can enter the IP address and mask in CIDR format, as follows.

```
device(config-if-1/1/1)# ip address 10.45.6.1/24
```

Syntax: no ip address ip- addr ip-mask [ospf-ignore | ospf-passive | secondary]

or

Syntax: no ip address ip-addr/mask-bits [ospf-ignore | ospf-passive | secondary]

The **ospf-ignore** and **ospf-passive** parameters modify the Layer 3 switch defaults for adjacency formation and interface advertisement. Use one of these parameters if you are configuring multiple IP subnet addresses on the interface but you want to prevent OSPF from running on some of the subnets:

- **ospf-passive** - This option disables adjacency formation with OSPF neighbors. By default, when OSPF is enabled on an interface, the software forms OSPF router adjacencies between each primary IP address on the interface and the OSPF neighbor attached to the interface.
- **ospf-ignore** - This option disables OSPF adjacency formation and also disables advertisement of the interface into OSPF. The subnet is completely ignored by OSPF.

NOTE

The **ospf-passive** option disables adjacency formation but does not disable advertisement of the interface into OSPF. To disable advertisement in addition to disabling adjacency formation, you must use the **ospf-ignore** option.

Use the **secondary** parameter if you have already configured an IP address within the same subnet on the interface.

NOTE

When you configure more than one address in the same subnet, all but the first address are secondary addresses and do not form OSPF adjacencies.

NOTE

All physical IP interfaces on Brocade FastIron Layer 3 devices share the same MAC address. For this reason, if more than one connection is made between two devices, one of which is a Brocade FastIron Layer 3 device, Brocade recommends the use of virtual interfaces. It is not recommended to connect two or more physical IP interfaces between two routers.

Assigning an IP address to a loopback interface

Loopback interfaces are always up, regardless of the states of physical interfaces. They can add stability to the network because they are not subject to route flap problems that can occur due to unstable links between a Layer 3 switch and other devices. You can configure up to eight loopback interfaces on a chassis Layer 3 switch devices. You can configure up to four loopback interfaces on a compact Layer 3 switch.

You can add up to 24 IP addresses to each loopback interface.

NOTE

If you configure the Brocade Layer 3 switch to use a loopback interface to communicate with a BGP4 neighbor, you also must configure a loopback interface on the neighbor and configure the neighbor to use that loopback interface to communicate with the Brocade Layer 3 switch. Refer to [Assigning an IP address to a loopback interface](#).

To add a loopback interface, enter commands such as those shown in the following example.

```
device(config-bgp-router)# exit
device(config)# interface loopback 1
device(config-lbif-1)# ip address 10.0.0.1/24
```

Syntax: interface loopback *num*

The *num* parameter specifies the virtual interface number. You can specify from 1 to the maximum number of virtual interfaces supported on the device. To display the maximum number of virtual interfaces supported on the device, enter the **show default values** command. The maximum is listed in the System Parameters section, in the Current column of the virtual-interface row.

Assigning an IP address to a virtual interface

A virtual interface is a logical port associated with a Layer 3 Virtual LAN (VLAN) configured on a Layer 3 switch. You can configure routing parameters on the virtual interface to enable the Layer 3 switch to route protocol traffic from one Layer 3 VLAN to the other, without using an external router.

NOTE

The Brocade feature that allows routing between VLANs within the same device, without the need for external routers, is called Integrated Switch Routing (ISR).

You can configure IP routing interface parameters on a virtual interface. This section describes how to configure an IP address on a virtual interface. Other sections in this chapter that describe how to configure interface parameters also apply to virtual interfaces.

NOTE

The Layer 3 switch uses the lowest MAC address on the device (the MAC address of port 1 or 1/1/1) as the MAC address for all ports within all virtual interfaces you configure on the device.

To add a virtual interface to a VLAN and configure an IP address on the interface, enter commands such as the following.

```
device(config)# vlan 2 name IP-Subnet_10.1.2.0/24
device(config-vlan-2)# untag ethernet 1 to 4
device(config-vlan-2)# router-interface ve 1
device(config-vlan-2)# interface ve 1
device(config-vif-1)# ip address 10.1.2.1/24
```

The first two commands in this example create a Layer 3 protocol-based VLAN name "IP-Subnet_10.1.2.0/24" and add a range of untagged ports to the VLAN. The **router-interface** command creates virtual interface 1 as the routing interface for the VLAN.

Syntax: router-interface ve num

The *num* variable specifies the virtual interface number. You can enter a number from 1 through 4095.

When configuring virtual routing interfaces on a device, you can specify a number from 1 through 4095. However, the total number of virtual routing interfaces that are configured must not exceed the system-max limit of 512 (or 255 for the ICX 7250).

The last two commands move the configuration to the interface configuration mode for the virtual interface and assign an IP address to the interface.

Syntax: interface ve num

Configuring IP follow on a virtual routing interface

IP Follow allows multiple virtual routing interfaces to share the same IP address. With this feature, one virtual routing interface is configured with an IP address, while the other virtual routing interfaces are configured to use that IP address, thus, they "follow" the virtual routing interface that has the IP address. This feature is helpful in conserving IP address space.

Configuration limitations and feature limitations for IP Follow on a virtual routing interface

- When configuring IP Follow, the primary virtual routing interface should not have ACL or DoS Protection configured. It is recommended that you create a dummy virtual routing interface as the primary and use the IP-follow virtual routing interface for the network.
- Global Policy Based Routing is not supported when IP Follow is configured.
- IPv6 is not supported with IP Follow.
- FastIron devices support IP Follow with OSPF and VRRP protocols only.

Configuration syntax for IP Follow on a virtual routing interface

Configure IP Follow by entering commands such as the following.

```
device(config)# vlan 2 name IP-Subnet 10.1.2.0/24
device(config-vlan-2)# untag ethernet 1 to 4
device(config-vlan-2)# router-interface ve 1
device(config-vlan-2)# interface ve 1
device(config-vif-1)# ip address 10.10.2.1/24
device(config-vif-1)# interface ve 2
device(config-vif-2)# ip follow ve 1
device(config-vif-2)# interface ve 3
device(config-vif-3)# ip follow ve 1
```

Syntax: `[no] ip follow ve number`

For *number*, enter the ID of the virtual routing interface.

Use the **no** form of the command to disable the configuration.

Virtual routing interface 2 and 3 do not have their own IP subnet addresses, but share the IP address of virtual routing interface 1.

Deleting an IP address

To delete an IP address, enter the **no ip address** command.

```
device(config-if-e1000-1)# no ip address 10.1.2.1
```

This command deletes IP address 10.1.2.1. You do not need to enter the subnet mask.

To delete all IP addresses from an interface, enter the **no ip address *** command.

```
device(config-if-e1000-1)# no ip address *
```

Syntax: `[no] ip address ip-addr | *`

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

NOTE

31-bit subnet masks are supported on ICX 7250, ICX 7450, and ICX 7750 devices running the full Layer 3 image.

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

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

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

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

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

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

Configuring an IPv4 address with a 31-bit subnet mask

To configure an IPv4 address with a 31-bit subnet mask, enter the following commands.

You can configure an IPv4 address with a 31-bit subnet mask on any interface (for example, Ethernet, loopback, VE, or tunnel interfaces).

```
device(config)# interface ethernet 1/1/5
device(config-if-e1000-1/1/5)# ip address 10.9.9.9 255.255.255.254
```

You can also enter the IP address and mask in the Classless Inter-domain Routing (CIDR) format, as follows.

```
device(config-if-e1000-1/1/5)# ip address 10.9.9.9/31
```

Syntax: [no] ip address *ip-address ip-mask*

Syntax: [no] ip address *ip-address/subnet-mask-bits*

The *ip-address* variable specifies the host address. The *ip-mask* variable specifies the IP network mask. The *subnet-mask-bits* variable specifies the network prefix mask.

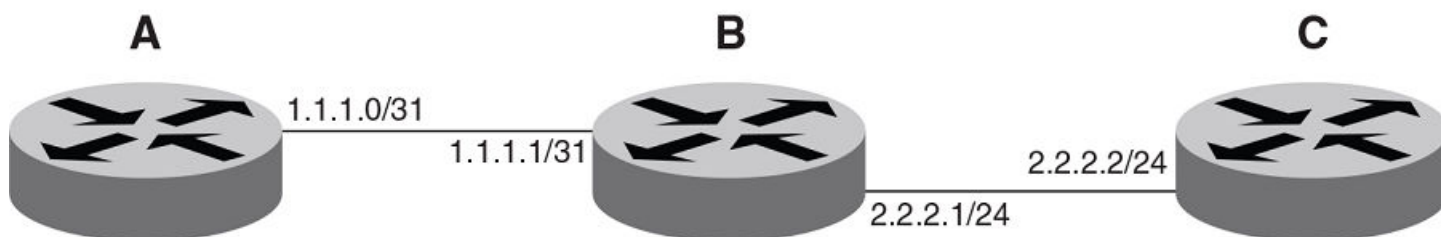
To disable configuration for an IPv4 address with a 31-bit subnet mask on any interface, use the **no** form of the command.

You cannot configure a secondary IPv4 address with a 31-bit subnet mask on any interface. The following error message is displayed when a secondary IPv4 address with a 31-bit subnet mask is configured.

```
Error: Cannot assign /31 subnet address as secondary
```

Configuration example

FIGURE 4 Configured 31-bit and 24-bit subnet masks



Router A is connected to Router B as a point-to-point link with 10.1.1.0/31 subnet. There are only two available addresses in this subnet, 10.1.1.0 on Router A and 10.1.1.1 on Router B.

Routers B and C are connected by a regular 24-bit subnet. Router C can either be a switch with many hosts belonging to the 10.2.2.2/24 subnet connected to it, or it can be a router.

Router A

```
RouterA(config)# interface ethernet 1/1/1
RouterA(config-if-e1000-1/1/1)# ip address 10.1.1.0/31
```

Router B

```
RouterB(config)# interface ethernet 1/1/1
RouterB(config-if-e1000-1/1/1)# ip address 10.1.1.1/31
RouterB(config-if-e1000-1/1/1)# exit
RouterB(config)# interface ethernet 1/3/1
RouterB(config-if-e1000-1/3/1)# ip address 10.2.2.1/24
```

Router C

```
RouterC(config# interface ethernet 1/3/1
RouterC(config-if-e1000-1/3/1)# ip address 10.2.2.2/24
```

Displaying information for a 31-bit subnet mask

Use the following commands to display information for the 31-bit subnet mask:

- **show run interface**
- **show ip route**
- **show ip cache**

Configuring DNS resolver

The Domain Name System (DNS) resolver is a feature in a Layer 2 or Layer 3 switch 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.

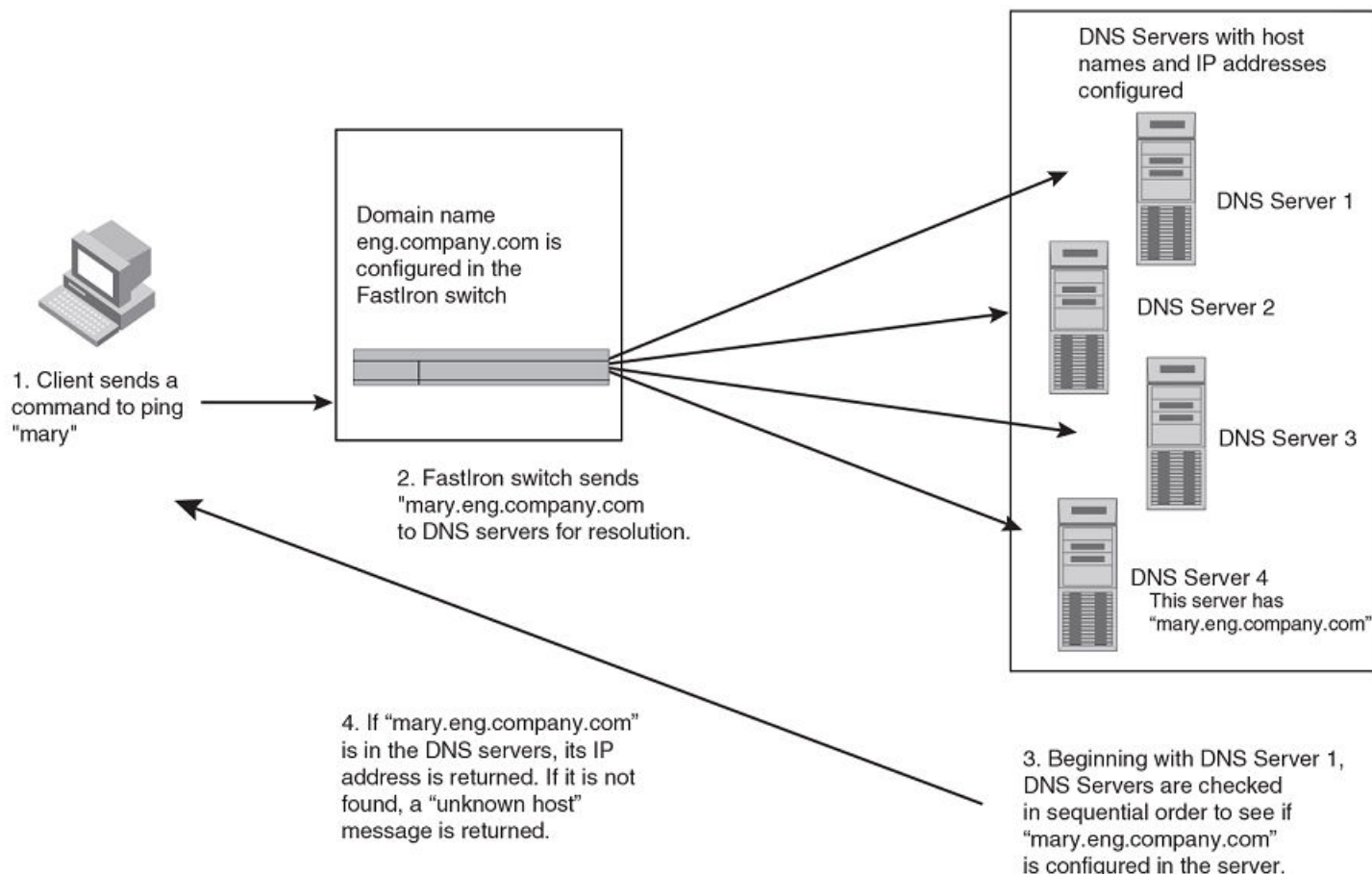
After you define a domain name, the Brocade device automatically appends the appropriate domain to a host and forwards it to the DNS servers for resolution.

For example, if the domain "ds.company.com" is defined on a Layer 2 or Layer 3 switch and you want to initiate a ping to "mary", you must reference only the host name instead of the host name and its domain name. For example, you could enter the following command to initiate the ping.

```
Brocade:> ping mary
```

The Layer 2 or Layer 3 switch qualifies the host name by appending a domain name (for example, mary.ds1.company.com). This qualified name is sent to the DNS server for resolution. If there are four DNS servers configured, it is sent to the first DNS server. If the host name is not resolved, it is sent to the second DNS server. If a match is found, a response is sent back to the client with the host IP address. If no match is found, an "unknown host" message is returned.

FIGURE 5 DNS resolution with one domain name



Defining DNS server addresses

You can configure the Brocade device to recognize up to four DNS servers. The first entry serves as the primary default address. If a query to the primary address fails to be resolved after three attempts, the next DNS address is queried (also up to three times). This process continues for each defined DNS address until the query is resolved. The order in which the default DNS addresses are polled is the same as the order in which you enter them.

To define DNS servers, enter the **ip dns server-address** command.

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

Syntax: [no] **ip dns server-address** *ip-addr* [*ip-addr*] [*ip-addr*] [*ip-addr*]

In this example, the first IP address entered becomes the primary DNS address and all others are secondary addresses. Because IP address 10.98.7.15 is the last address listed, it is also the last address consulted to resolve a query.

Defining a domain list

If you want to use more than one domain name to resolve host names, you can create a list of domain names. For example, enter the commands such as the following.

```
device(config)# ip dns domain-list company.com
device(config)# ip dns domain-list ds.company.com
device(config)# ip dns domain-list hw_company.com
device(config)# ip dns domain-list qa_company.com
```

The domain names are tried in the order you enter them.

Syntax: [no] ip dns domain-list *domain-name*

Using a DNS name to initiate a trace route

Suppose you want to trace the route from a Brocade Layer 3 switch to a remote server identified as NYC02 on domain newyork.com. Because the NYC02@ds1.newyork.com domain is already defined on the Layer 3 switch, you need to enter only the host name, NYC02, as noted in the following example.

```
device# traceroute nyc02
```

The only required parameter is the IP address of the host at the other end of the route.

After you enter the **traceroute** command, a message indicating that the DNS query is in process and the current gateway address (IP address of the domain name server) being queried appear on the screen. When traceroute fails, an error occurs as shown in the last two lines in the following example.

```
Type Control-c to abort
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
Trace route to target IP node 10.157.22.80 failed.
IP: Errno(9) No response from target or intermediate node
```

NOTE

In the example, 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 NYC02 host.

Configuring packet parameters

You can configure the following packet parameters on Layer 3 switches. These parameters control how the Layer 3 switch sends IP packets to other devices on an Ethernet network. The Layer 3 switch always places IP packets into Ethernet packets to forward them on an Ethernet port.

- Encapsulation type - The format for the Layer 2 packets within which the Layer 3 switch sends IP packets.
- Maximum Transmission Unit (MTU) - The maximum length of IP packet that a Layer 2 packet can contain. IP packets that are longer than the MTU are fragmented and sent in multiple Layer 2 packets. You can change the MTU globally or an individual ports:
 - Global MTU - The default MTU value depends on the encapsulation type on a port and is 1500 bytes for Ethernet II encapsulation and 1492 bytes for SNAP encapsulation.
 - Port MTU - A port default MTU depends on the encapsulation type enabled on the port.

Changing the encapsulation type

The Layer 3 switch encapsulates IP packets into Layer 2 packets, to send the IP packets on the network. (A Layer 2 packet is also called a MAC layer packet or an Ethernet frame.) The source address of a Layer 2 packet is the MAC address of the Layer 3 switch interface sending the packet. The destination address can be one of the following:

- The MAC address of the IP packet destination. In this case, the destination device is directly connected to the Layer 3 switch.
- The MAC address of the next-hop gateway toward the packet destination.
- An Ethernet broadcast address.

The entire IP packet, including the source and destination address and other control information and the data, is placed in the data portion of the Layer 2 packet. Typically, an Ethernet network uses one of two different formats of Layer 2 packet:

- Ethernet II
- Ethernet SNAP (also called IEEE 802.3)

The control portions of these packets differ slightly. All IP devices on an Ethernet network must use the same format. Brocade Layer 3 switches use Ethernet II by default. You can change the IP encapsulation to Ethernet SNAP on individual ports if needed.

NOTE

All devices connected to the Layer 3 switch port must use the same encapsulation type.

To change the IP encapsulation type on interface 5 to Ethernet SNAP, enter the following commands.

```
device(config)# interface ethernet 5
device(config-if-e1000-5)# ip encapsulation snap
```

Syntax: `ip encapsulation { snap | ethernet_ii }`

Changing the MTU

The Maximum Transmission Unit (MTU) is the maximum length of IP packet that a Layer 2 packet can contain. IP packets that are longer than the MTU are fragmented and sent in multiple Layer 2 packets. You can change the MTU globally or on individual ports.

The default MTU is 1500 bytes for Ethernet II packets and 1492 for Ethernet SNAP packets.

MTU enhancements

Brocade devices contain the following enhancements to jumbo packet support:

- Hardware forwarding of Layer 3 jumbo packets - Layer 3 IP unicast jumbo packets received on a port that supports the frame MTU size and forwarded to another port that also supports the frame MTU size are forwarded in hardware. Previous releases support hardware forwarding of Layer 2 jumbo frames only.
- ICMP unreachable message if a frame is too large to be forwarded - If a jumbo packet has the Do not Fragment (DF) bit set, and the outbound interface does not support the packet MTU size, the Brocade device sends an ICMP unreachable message to the device that sent the packet.

NOTE

These enhancements apply only to transit traffic forwarded through the Brocade device.

Configuration considerations for increasing the MTU

- The MTU command is applicable to VEs and physical IP interfaces. It applies to traffic routed between networks.

- For ICX 7250, ICX 7450, and ICX 7750 devices, the IPv4 and IPv6 MTU values are the same. Modifying one also changes the value of the other.
- For ICX 7250, ICX 7450, and ICX 7750 devices, the minimum IPv4 and IPv6 MTU values for both physical and virtual interfaces are 1280.
- You cannot use this command to set Layer 2 maximum frame sizes per interface. The global **jumbo** command causes all interfaces to accept Layer 2 frames.
- When you increase the MTU size of a port, the increase uses system resources. Increase the MTU size only on the ports that need it. For example, if you have one port connected to a server that uses jumbo frames and two other ports connected to clients that can support the jumbo frames, increase the MTU only on those three ports. Leave the MTU size on the other ports at the default value (1500 bytes). Globally increase the MTU size only if needed.

Forwarding traffic to a port with a smaller MTU size

In order to forward traffic from a port with 1500 MTU configured to a port that has a smaller MTU (for example, 750) size, you must apply the **mtu-exceed forward** global command. To remove this setting, enter the **mtu-exceed hard-drop** command. The **hard-drop** option is enabled by default on the router.

Syntax: **mtu-exceed { forward | hard-drop }**

- **forward**—Fragments and forwards a packet from a port with a larger MTU to a port with a smaller MTU.
- **hard-drop**—Resets to default and removes the forward function.

Globally changing the Maximum Transmission Unit

The Maximum Transmission Unit (MTU) is the maximum size an IP packet can be when encapsulated in a Layer 2 packet. If an IP packet is larger than the MTU allowed by the Layer 2 packet, the Layer 3 switch fragments the IP packet into multiple parts that will fit into the Layer 2 packets, and sends the parts of the fragmented IP packet separately, in different Layer 2 packets. The device that receives the multiple fragments of the IP packet reassembles the fragments into the original packet.

You can increase the MTU size to accommodate jumbo packet sizes up to 10,200 bytes.

To globally enable jumbo support on all ports of a FastIron device, enter commands such as the following.

```
device(config)# jumbo
device(config)# write memory
device(config)# end
device# reload
```

Syntax: **[no] jumbo**

NOTE

You must save the configuration change and then reload the software to enable jumbo support.

Changing the MTU on an individual port

By default, the maximum Ethernet MTU sizes are as follows:

- 1500 bytes - The maximum for Ethernet II encapsulation
- 1492 bytes - The maximum for SNAP encapsulation

When jumbo mode is enabled, the maximum Ethernet MTU sizes are as follows:

- 10,218 bytes - The maximum for Ethernet II encapsulation (Default MTU: 9216)

- 10,214 bytes - The maximum for SNAP encapsulation (Default MTU: 9216)

NOTE

If you set the MTU of a port to a value lower than the global MTU and from 576 through 1499, the port fragments the packets. However, if the port MTU is exactly 1500 and this is larger than the global MTU, the port drops the packets. For ICX 7250, ICX 7450, and ICX 7750 devices, the minimum IPv4 and IPv6 MTU values for both physical and virtual interfaces are 1280.

NOTE

You must save the configuration change and then reload the software to enable jumbo support.

To change the MTU for interface 1/1/5 to 1000, enter the following commands.

```
device(config)# interface ethernet 1/1/5
device(config-if-1/1/5)# ip mtu 1000
device(config-if-1/1/5)# write memory
device(config-if-1/1/5)# end
device# reload
```

Syntax: [no] ip mtu *num*

The *num* variable specifies the MTU. Ethernet II packets can hold IP packets from 576 through 1500 bytes long. If jumbo mode is enabled, Ethernet II packets can hold IP packets up to 10,218 bytes long. Ethernet SNAP packets can hold IP packets from 576 through 1492 bytes long. If jumbo mode is enabled, SNAP packets can hold IP packets up to 10,214 bytes long. The default MTU for Ethernet II packets is 1500. The default MTU for SNAP packets is 1492.

Path MTU discovery (RFC 1191) support

ICX 7250, ICX 7450, and ICX 7750 devices support the path MTU discovery method described in RFC 1191. When the Brocade device receives an IP packet that has its Do not Fragment (DF) bit set, and the packet size is greater than the MTU value of the outbound interface, then the Brocade device returns an ICMP Destination Unreachable message to the source of the packet, with the Code indicating "fragmentation needed and DF set". The ICMP Destination Unreachable message includes the MTU of the outbound interface. The source host can use this information to help determine the maximum MTU of a path to a destination.

RFC 1191 is supported on all interfaces.

Changing the router ID

In most configurations, a Layer 3 switch has multiple IP addresses, usually configured on different interfaces. As a result, a Layer 3 switch identity to other devices varies depending on the interface to which the other device is attached. Some routing protocols, including Open Shortest Path First (OSPF) and Border Gateway Protocol version 4 (BGP4), identify a Layer 3 switch by just one of the IP addresses configured on the Layer 3 switch, regardless of the interfaces that connect the Layer 3 switches. This IP address is the router ID.

NOTE

Routing Information Protocol (RIP) does not use the router ID.

NOTE

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

By default, the router ID on a Brocade Layer 3 switch is one of the following:

- If the router has loopback interfaces, the default router ID is the IP address configured on the lowest numbered loopback interface configured on the Layer 3 switch. For example, if you configure loopback interfaces 1, 2, and 3 as follows, the default router ID is 10.9.9.9/24:
 - Loopback interface 1, 10.9.9.9/24

- Loopback interface 2, 10.4.4.4/24
- Loopback interface 3, 10.1.1.1/24
- If the device does not have any loopback interfaces, the default router ID is the lowest numbered IP interface configured on the device.

If you prefer, you can explicitly set the router ID to any valid IP address. The IP address cannot be in use on another device in the network.

NOTE

Brocade Layer 3 switches use the same router ID for both OSPF and BGP4. If the router is already configured for OSPF, you may want to use the router ID that is already in use on the router rather than set a new one. To display the router ID, enter the **show ip** command at any CLI level or select the IP->General links from the Configure tree in the Web Management Interface.

To change the router ID, enter a command such as the following.

```
device(config)# ip router-id 10.157.22.26
```

Syntax: **ip router-id** ip-addr

The *ip-addr* variable can be any valid, unique IP address.

NOTE

You can specify an IP address used for an interface on the Brocade Layer 3 switch, but do not specify an IP address in use by another device.

Specifying a single source interface for specified packet types

NOTE

This feature is supported on the ICX 7750 switch.

When the Layer 3 switch originates a packet of one of the following types, the 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
- SNMP traps

You can configure the Layer 3 switch 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 Layer 3 switch uses the same IP address as the source for all packets of the specified type, regardless of the ports that actually sends the packets.

Identifying a single source IP address for specified packets provides the following benefits:

- If your server is configured to accept packets only from specific IP addresses, you can use this feature to simplify configuration of the server by configuring the Brocade 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 the client or server 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.

The software contains separate CLI commands for specifying the source interface for specific packets. You can configure a source interface for one or more of these types of packets separately.

The following sections show the syntax for specifying a single source IP address for specific packet types.

Telnet packets

To specify the lowest-numbered IP address configured on a virtual interface as the device source for all Telnet packets, enter commands such as the following.

```
device(config)# interface loopback 2
device(config-lbif-2)# ip address 10.0.0.2/24
device(config-lbif-2)# exit
device(config)# ip telnet source-interface loopback 2
```

The commands in this example configure loopback interface 2, assign IP address 10.0.0.2/24 to the interface, then designate the interface as the source for all Telnet packets from the Layer 3 switch.

The following commands configure an IP interface on an Ethernet port and designate the address port as the source for all Telnet packets from the Layer 3 switch.

```
device(config)# interface ethernet 1/1/4
device(config-if-1/1/4)# ip address 10.157.22.110/24
device(config-if-1/1/4)# exit
device(config)# ip telnet source-interface ethernet 1/1/4
```

Syntax: [no] ip telnet source-interface { ethernet unit / slot / port | loopback num | management num | venum }

TACACS/TACACS+ packets

To specify the lowest-numbered IP address configured on a virtual interface as the device source for all TACACS/TACACS+ packets, enter commands such as the following.

```
device(config)# interface ve 1
device(config-vif-1)# ip address 10.0.0.3/24
device(config-vif-1)# exit
device(config)# ip tacacs source-interface ve 1
```

The commands in this example configure virtual interface 1, assign IP address 10.0.0.3/24 to the interface, then designate the interface as the source for all TACACS/TACACS+ packets from the Layer 3 switch.

Syntax: [no] ip tacacs source-interface { ethernet unit / slot / port | loopback num | management num | venum }

RADIUS packets

To specify the lowest-numbered IP address configured on a virtual interface as the device source for all RADIUS packets, enter commands such as the following.

```
device(config)# interface ve 1
device(config-vif-1)# ip address 10.0.0.3/24
device(config-vif-1)# exit
device(config)# ip radius source-interface ve 1
```

The commands in this example configure virtual interface 1, assign IP address 10.0.0.3/24 to the interface, then designate the interface as the source for all RADIUS packets from the Layer 3 switch.

Syntax: [no] ip radius source-interface { ethernet unit / slot / port | loopback num | management num | venum }

TFTP packets

To specify the lowest-numbered IP address configured on a virtual interface as the device source for all TFTP packets, enter commands such as the following.

```
device(config)# interface ve 1
device(config-vif-1)# ip address 10.0.0.3/24
device(config-vif-1)# exit
device(config)# ip tftp source-interface ve 1
```

The commands in this example configure virtual interface 1, assign IP address 10.0.0.3/24 to the interface, then designate the interface's address as the source address for all TFTP packets.

Syntax: `[no] ip tftp source-interface { ethernet unit / slot / port | loopback num | management num | venum }`

The default is the lowest-numbered IP address configured on the port through which the packet is sent. The address therefore changes, by default, depending on the port.

Syslog packets

To specify the lowest-numbered IP address configured on a virtual interface as the device source for all Syslog packets, enter commands such as the following.

```
device(config)# interface ve 1
device(config-vif-1)# ip address 10.0.0.4/24
device(config-vif-1)# exit
device(config)# ip syslog source-interface ve 1
```

The commands in this example configure virtual interface 1, assign IP address 10.0.0.4/24 to the interface, then designate the interface's address as the source address for all Syslog packets.

Syntax: `[no] ip syslog source-interface { ethernet unit / slot / port | loopback num | management num | venum }`

The default is the lowest-numbered IP or IPv6 address configured on the port through which the packet is sent. The address therefore changes, by default, depending on the port.

SNTP packets

To specify the lowest-numbered IP address configured on a virtual interface as the device source for all SNTP packets, enter commands such as the following.

```
device(config)# interface ve 1
device(config-vif-1)# ip address 10.0.0.5/24
device(config-vif-1)# exit
device(config)# ip sntp source-interface ve 1
```

The commands in this example configure virtual interface 1, assign IP address 10.0.0.5/24 to the interface, then designate the interface's address as the source address for all SNTP packets.

Syntax: `[no] ip sntp source-interface { ethernet unit / slot / port | loopback num | management num | venum }`

The default is the lowest-numbered IP or IPv6 address configured on the port through which the packet is sent. The address therefore changes, by default, depending on the port.

SNMP packets

To specify a loopback interface as the SNMP single source trap, enter commands such as the following.

```
device(config)# interface loopback 1
device(config-lbif-1)# ip address 10.0.0.1/24
```

```
device(config-lbif-1)# exit
device(config)# snmp-server trap-source loopback 1
```

The commands in this example configure loopback interface 1, assign IP address 10.00.1/24 to the loopback interface, then designate the interface as the SNMP trap source for this device. Regardless of the port the Brocade device uses to send traps to the receiver, the traps always arrive from the same source IP address.

Syntax: [no] snmp-server trap-source { ethernet *unit / slot / port* | loopback *num* | venum }

Configuring delay time for notifying VE down event

When all the ports in the VLAN go into an inactive state (for example, the non-forwarding state), the device notifies the Layer 3 protocols of the VE down event only after the configured timer expires. Once the timer expires, the device checks if any of the ports is in the forwarding state. If no ports are in the forwarding state, the device notifies the Layer 3 protocols of the VE down event. If any of the ports is in the forwarding state, the device ignores the down event.

While the timer is running, if any of the ports comes into forwarding state, the device cancels the timer and does not notify the VE down event to the protocols.

NOTE

In the case of multiple flaps, if any of the ports comes into forwarding state before the delay notification timer expiry then the device cancels the timer and a fresh timer is started during port down event. In case of continuous flaps where flap time is less than delay notification timer, the flaps can be detected by other methods like port statistics or drop in traffic or by the convergence logs of layer2 loop detection protocols.

Suppressing the link status notification allows a quick port status change and recovery to occur without triggering any of the changes that are necessary when a port stays down.

By default, the delay time is not configured.

NOTE

Configuring delayed Layer 3 notifications on the VE feature is supported on ICX 7250, ICX 7450, and ICX 7750. product families from Brocade.

Configuring VE down time notification

Perform the following steps to configure the delay time for notifying the Layer 3 protocols of the VE down event.

1. From global configuration mode, enter VE interface configuration mode.

```
device(config)# interface ve 50
```

2. Configure the delay notifications time value.

```
device(config-vif-50)# delay-notifications 20
```

3. Use the **show ip interface ve** command to confirm the configuration.

The following example shows how to configure the delay time for notifying the Layer 3 protocols of the VE down event.

```
device(config)# interface ve 50
device(config-vif-50)# delay-notifications 20
```

Configuring forwarding parameters

The following configurable parameters control the forwarding behavior of Brocade Layer 3 switches:

- Time-To-Live (TTL) threshold
- Forwarding of directed broadcasts
- Forwarding of source-routed packets
- Ones-based and zero-based broadcasts

All these parameters are global and thus affect all IP interfaces configured on the Layer 3 switch.

Changing the TTL threshold

The time to live (TTL) threshold prevents routing loops by specifying the maximum number of router hops an IP packet originated by the Layer 3 switch can travel through. Each device capable of forwarding IP that receives the packet decrements (decreases) the packet TTL by one. If a device receives a packet with a TTL of 1 and reduces the TTL to zero, the device drops the packet.

The default value for the TTL threshold is 64. You can change the TTL threshold to a value from 1 through 255.

To modify the TTL threshold to 25, enter the **ip ttl** command.

```
device(config)# ip ttl 25
```

Syntax: **ip ttl** *ttl-threshold*

Enabling forwarding of directed broadcasts

A directed broadcast is an IP broadcast to all devices within a single directly-attached network or subnet. A net-directed broadcast goes to all devices on a given network. A subnet-directed broadcast goes to all devices within a given subnet.

NOTE

A less common type, the all-subnets broadcast, goes to all directly-attached subnets. Forwarding for this broadcast type also is supported, but most networks use IP multicasting instead of all-subnet broadcasting.

Forwarding for all types of IP directed broadcasts is disabled by default. You can enable forwarding for all types if needed. You cannot enable forwarding for specific broadcast types.

To enable forwarding of IP directed broadcasts, enter the **ip directed-broadcast** command in device configuration mode.

```
device # configure terminal
device(config)# ip directed-broadcast
```

Syntax: **[no] ip directed-broadcast**

Brocade software makes the forwarding decision based on the router's knowledge of the destination network prefix. Routers cannot determine that a message is unicast or directed broadcast apart from the destination network prefix. The decision to forward or not forward the message is by definition only possible in the last hop router.

To disable the directed broadcasts, enter the **no ip directed-broadcast** command in device configuration mode.

```
device # configure terminal
device(config)# no ip directed-broadcast
```

To enable directed broadcasts on an individual interface instead of globally for all interfaces, enter the **ip directed-broadcast** command at the interface configuration level as shown in the following example.

```
device # configure terminal
device(config)# interface ethernet 1/1/1
device(config-if-1/1/1 # ip directed-broadcast
```

Disabling forwarding of IP source-routed packets

A source-routed packet specifies the exact router path for the packet. The packet specifies the path by listing the IP addresses of the router interfaces through which the packet must pass on its way to the destination. The Layer 3 switch supports both types of IP source routing:

- Strict source routing - Requires the packet to pass through only the listed routers. If the Layer 3 switch receives a strict source-routed packet but cannot reach the next hop interface specified by the packet, the Layer 3 switch discards the packet and sends an ICMP Source-Route-Failure message to the sender.

NOTE

The Layer 3 switch allows you to disable sending of the Source-Route-Failure messages.

- Loose source routing - Requires that the packet pass through all of the listed routers but also allows the packet to travel through other routers, which are not listed in the packet.

The Layer 3 switch forwards both types of source-routed packets by default. To disable the feature, use either of the following methods. You cannot enable or disable strict or loose source routing separately.

To disable forwarding of IP source-routed packets, enter the **no ip source-route** command.

```
device # configure terminal
device(config)# no ip source-route
```

Syntax: [no] ip source-route

To re-enable forwarding of source-routed packets, enter the **ip source-route** command.

```
device # configure terminal
device(config)# ip source-route
```

Enabling support for zero-based IP subnet broadcasts

By default, the Layer 3 switch treats IP packets with all ones in the host portion of the address as IP broadcast packets. For example, the Layer 3 switch treats IP packets with 10.157.22.255/24 as the destination IP address as IP broadcast packets and forwards the packets to all IP hosts within the 10.157.22.x subnet (except the host that sent the broadcast packet to the Layer 3 switch).

Most IP hosts are configured to receive IP subnet broadcast packets with all ones in the host portion of the address. However, some older IP hosts instead expect IP subnet broadcast packets that have all zeros instead of all ones in the host portion of the address. To accommodate this type of host, you can enable the Layer 3 switch to treat IP packets with all zeros in the host portion of the destination IP address as broadcast packets.

NOTE

When you enable the Layer 3 switch for zero-based subnet broadcasts, the Layer 3 switch still treats IP packets with all ones in the host portion as IP subnet broadcasts too. Thus, the Layer 3 switch can be configured to support all ones only (the default) or all ones and all zeroes.

NOTE

This feature applies only to IP subnet broadcasts, not to local network broadcasts. The local network broadcast address is still expected to be all ones.

To enable the Layer 3 switch for zero-based IP subnet broadcasts in addition to ones-based IP subnet broadcasts, enter the following command.

```
device(config)# ip broadcast-zero
device(config)# write memory
device(config)# end
device# reload
```

NOTE

You must save the configuration and reload the software to place this configuration change into effect.

Syntax: [no] ip broadcast-zero

Disabling ICMP messages

Brocade devices are enabled to reply to ICMP echo messages and send ICMP Destination Unreachable messages by default.

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

- **Echo messages (ping messages)** - The Layer 3 switch replies to IP pings from other IP devices.
- **Destination Unreachable messages** - If the Layer 3 switch receives an IP packet that it cannot deliver to its destination, the Layer 3 switch discards the packet and sends a message back to the device that sent the packet to the Layer 3 switch. The message informs the device that the destination cannot be reached by the Layer 3 switch.

Disabling replies to broadcast ping requests

By default, Brocade devices are enabled to respond to broadcast ICMP echo packets, which are ping requests.

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

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

Syntax: [no] ip icmp echo broadcast-request

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

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

Disabling ICMP destination unreachable messages

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

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

- **Source-route-fail** - The device received a source-routed packet but cannot locate the next-hop IP address indicated in the packet Source-Route option.

You can disable the Brocade device from sending these types of ICMP messages on an individual basis. To do so, use the following CLI method.

NOTE

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

To disable all ICMP Unreachable messages, enter the **no ip icmp unreachable** command.

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

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

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

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

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

If you have disabled all ICMP Unreachable message types but you want to re-enable certain types, for example ICMP Host Unreachable messages, you can do so by entering the following command.

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

Enabling ICMP redirect messages

You can enable and disable IPv4 ICMP redirect messages globally or on individual Virtual Ethernet (VE) interfaces but not on individual physical interfaces.

NOTE

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

By default, IP ICMP redirect over global level is disabled and a Brocade Layer 3 switch does not send an ICMP redirect message to the source of a misdirected packet in addition to forwarding the packet to the appropriate router. To enable ICMP redirect messages globally, enter the following command at the global CONFIG level of the CLI:

```
device(config)# ip icmp redirect
```

Syntax: **[no] ip icmp redirect**

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

```
Brocade(config-vlan-10)# interface ve 10
Brocade(config-vif-10)# no ip redirect
```

Syntax: [no] ip redirect

Configuring a default network route

The Layer 3 switch enables you to specify a candidate default route without the need to specify the next hop gateway. If the IP route table does not contain an explicit default route (for example, 0.0.0.0/0) or propagate an explicit default route through routing protocols, the software can use the default network route as a default route instead.

When the software uses the default network route, it also uses the default network route's next hop gateway as the gateway of last resort.

This feature is especially useful in environments where network topology changes can make the next hop gateway unreachable. This feature allows the Layer 3 switch to perform default routing even if the default network route's default gateway changes.

The feature thus differs from standard default routes. When you configure a standard default route, you also specify the next hop gateway. If a topology change makes the gateway unreachable, the default route becomes unusable.

For example, if you configure 10.10.10.0/24 as a candidate default network route, if the IP route table does not contain an explicit default route (0.0.0.0/0), the software uses the default network route and automatically uses that route's next hop gateway as the default gateway. If a topology change occurs and as a result the default network route's next hop gateway changes, the software can still use the default network route. To configure a default network route, use the following CLI method.

If you configure more than one default network route, the Layer 3 switch uses the following algorithm to select one of the routes.

1. Use the route with the lowest administrative distance.
2. If the administrative distances are equal:
 - Are the routes from different routing protocols (RIP, OSPF, or BGP4)? If so, use the route with the lowest IP address.
 - If the routes are from the same routing protocol, use the route with the best metric. The meaning of "best" metric depends on the routing protocol:
 - **RIP** - The metric is the number of hops (additional routers) to the destination. The best route is the route with the fewest hops.
 - **OSPF** - The metric is the path cost associated with the route. The path cost does not indicate the number of hops but is instead a numeric value associated with each route. The best route is the route with the lowest path cost.
 - **BGP4** - The metric is the Multi-exit Discriminator (MED) associated with the route. The MED applies to routes that have multiple paths through the same Autonomous System. The best route is the route with the lowest MED.

Example of configuring a default network route

You can configure up to four default network routes.

To configure a default network route, enter commands such as the following.

```
device(config)# ip default-network 10.157.22.0
device(config)# write memory
```

Syntax: ip default-network *ip-addr*

The *ip-addr* variable specifies the network address.

To verify that the route is in the route table, enter the following command at any level of the CLI.

```
device# show ip route
Total number of IP routes: 2
Start index: 1  B:BGP D:Connected R:RIP S:Static O:OSPF *:Candidate default
  Destination      NetMask      Gateway      Port      Cost      Type
1      10.157.20.0    255.255.255.0  0.0.0.0      1b1        1        D
2      10.157.22.0    255.255.255.0  0.0.0.0      1/4/11     1        *D
```

This example shows two routes. Both of the routes are directly attached, as indicated in the Type column. However, one of the routes is shown as type "*D", with an asterisk (*). The asterisk indicates that this route is a candidate for the default network route.

Configuring IP load sharing

The IP route table can contain more than one path to a given destination. When this occurs, the Layer 3 switch selects the path with the lowest cost as the path for forwarding traffic to the destination. If the IP route table contains more than one path to a destination and the paths each have the lowest cost, then the Layer 3 switch uses IP load sharing to select a path to the destination.

IP load sharing uses a hashing algorithm based on the source IP address, destination IP address, and protocol field in the IP header, TCP, and UDP information.

NOTE

IP load sharing is also called "Equal-Cost Multi-Path (ECMP) load sharing or just ECMP.

NOTE

IP load sharing is based on next-hop routing, and not on source routing.

NOTE

The term "path" refers to the next-hop router to a destination, not to the entire route to a destination. Thus, when the software compares multiple equal-cost paths, the software is comparing paths that use different next-hop routers, with equal costs, to the same destination. In many contexts, the terms "route" and "path" mean the same thing. The term "path" is used in this section to refer to an individual next-hop router to a destination, while the term "route" refers collectively to the multiple paths to the destination. Load sharing applies when the IP route table contains multiple, equal-cost paths to a destination.

NOTE

Brocade devices also perform load sharing among the ports in aggregate links. Refer to "Trunk group load sharing" in the *Brocade FastIron Platform and Layer 2 Switching Configuration Guide*.

How multiple equal-cost paths enter the IP route table

IP load sharing applies to equal-cost paths in the IP route table. Routes that are eligible for load sharing can enter the routing table from any of the following routing protocols:

- IP static routes
- Routes learned through OSPF
- Routes learned through BGP4

Administrative distance for each IP route

The administrative distance is a unique value associated with each type (source) of IP route. Each path has an administrative distance. The administrative distance is not used when performing IP load sharing, but the administrative distance is used when evaluating multiple equal-cost paths to the same destination from different sources, such as between static IP routes, OSPF, and BGP4.

The value of the administrative distance is determined by the source of the route. The Layer 3 switch is configured with a unique administrative distance value for each IP route source.

When the software receives multiple paths to the same destination and the paths are from different sources, the software compares the administrative distances of the paths and selects the path with the lowest administrative distance. The software then places the path with the lowest administrative distance in the IP route table. For example, if the Layer 3 switch has a path learned from OSPF and a path learned from IBGP for a given destination, only the path with the lower administrative distance enters the IP route table.

Here are the default administrative distances on the Brocade Layer 3 switch:

- Directly connected - 0 (this value is not configurable)
- Static IP route - 1 (applies to all static routes, including default routes and default network routes)
- Exterior Border Gateway Protocol (EBGP) - 20
- OSPF - 110
- Interior Gateway Protocol (IBGP) - 200
- Local BGP - 200
- Unknown - 255 (the router will not use this route)

Lower administrative distances are preferred over higher distances. For example, if the router receives routes for the same network from OSPF and from IBGP, the router will prefer the OSPF route by default.

NOTE

You can change the administrative distances individually. Refer to the configuration chapter for the route source for information.

Since the software selects only the path with the lowest administrative distance, and the administrative distance is determined by the path source, IP load sharing applies only when the IP route table contains multiple paths to the same destination, from the same IP route source.

IP load sharing does not apply to paths that come from different sources.

Path cost

The cost parameter provides a common basis of comparison for selecting from among multiple paths to a given destination. Each path in the IP route table has a cost. When the IP route table contains multiple paths to a destination, the Layer 3 switch chooses the path with the lowest cost. When the IP route table contains more than one path with the lowest cost to a destination, the Layer 3 switch uses IP load sharing to select one of the lowest-cost paths.

The source of a path cost value depends on the source of the path:

- **IP static route** - The value you assign to the metric parameter when you configure the route. The default metric is 1.
- **OSPF** - The Path Cost associated with the path. The paths can come from any combination of inter-area, intra-area, and external Link State Advertisements (LSAs).
- **BGP4** - The path Multi-Exit Discriminator (MED) value.

NOTE

If the path is redistributed between two or more of the above sources before entering the IP route table, the cost can increase during the redistribution due to settings in redistribution filters.

Static route, OSPF, and BGP4 load sharing

IP load sharing and load sharing for BGP4 routes are individually configured. Multiple equal-cost paths for a destination can enter the IP route table only if the source of the paths is configured to support multiple equal-cost paths. For example, if BGP4 allows only one path with a given cost for a given destination, the BGP4 route table cannot contain equal-cost paths to the destination. Consequently, the IP route table will not receive multiple equal-cost paths from BGP4.

The load sharing state for all the route sources is based on the state of IP load sharing. Since IP load sharing is enabled by default on all Brocade Layer 3 switches, load sharing for static IP routes, OSPF routes, and BGP4 routes also is enabled by default.

NOTE

In the table below, the default and the maximum number of paths for a static IP route and OSPF depend on the value for IP load sharing, and are not separately configurable.

NOTE

In the table below, the default and the maximum number of paths are not applicable for BGP4 using the Brocade ICX 7250.

TABLE 7 Default load sharing parameters for route sources

Route source	Default maximum number of paths	Maximum number of paths	
		ICX 7450 / ICX 7250	ICX 7750
Static IP route	4	8	32
OSPF	4	8	32
BGP4	1	4	32

How IP load sharing works

When ECMP is enabled, multiple equal-cost paths for the destination IP is installed in the hardware Layer 3 routing table. When an ingress Layer 3 IP traffic matches with the entry in the hardware for Layer 3 routing, one of the paths is selected based on the internal Hardware hashing logic and the packet gets forwarded on that path.

Disabling IP load sharing

To disable IP load sharing, enter the following commands.

```
device(config)# no ip load-sharing
```

Syntax: no ip load-sharing

Changing the maximum number of ECMP (load sharing) paths

You can change the maximum number of paths the Layer 3 switch supports to a value from 2 through 8. On the Brocade ICX 7750, the value range for the maximum number of load-sharing paths is from 2 through 32.

TABLE 8 Maximum number of ECMP load sharing paths per device

ICX 7250 / ICX 7450	ICX 7750
8	32

For optimal results, set the maximum number of paths to a value at least as high as the maximum number of equal-cost paths your network typically contains. For example, if the Layer 3 switch you are configuring for IP load sharing has six next-hop routers, set the maximum paths value to six.

To change the number of IP load sharing paths, enter a command such as the following.

```
device(config)# ip load-sharing 6
```

Syntax: [no] ip load-sharing [*num*]

The *num* variable specifies the number of paths and can be from 2 through 8, depending on the device you are configuring. On the Brocade ICX 7750, the value of the *num* variable can be from 2 through 32.

The configuration of the maximum number of IP load sharing paths to a value more than 8 is determined by the maximum number of ECMP paths defined at the system level using the **system-max max-ecmp** command. You cannot configure the maximum number of IP load sharing paths higher than the value defined at the system level. Also, you cannot configure the maximum number of ECMP paths at the system level to a value less than the configured IP load sharing value.

To define the maximum number of ECMP paths at the system level, enter a command such as the following.

```
device(config)# system-max max-ecmp 20
device(config)# write memory
device(config)# exit
device# reload
```

Syntax: [no] system-max max-ecmp [*num*]

The *num* variable specifies the maximum number of ECMP paths and the value range can be from 8 through 32. This command is supported only on the Brocade ICX 7750.

You must save the configuration and reload the device for the maximum ECMP value change to take effect.

ECMP load sharing for IPv6

The IPv6 route table selects the best route to a given destination from among the routes in the tables maintained by the configured routing protocols (BGP4, OSPF, static, and so on). The IPv6 route table can contain more than one path to a given destination. When this occurs, the Brocade device selects the path with the lowest cost for insertion into the routing table. If more than one path with the lowest cost exists, all of these paths are inserted into the routing table, subject to the configured maximum number of load sharing paths (by default 4). The device uses Equal-Cost Multi-Path (ECMP) load sharing to select a path to a destination.

When a route is installed by routing protocols or configured static route for the first time, and the IPv6 route table contains multiple, equal-cost paths to that route, the device checks the IPv6 neighbor for each next hop. Every next hop where the link layer address has been resolved will be stored in hardware. The device will initiate neighbor discovery for the next hops whose link layer addresses are not resolved. The hardware will hash the packet and choose one of the paths. The number of paths would be updated in hardware as the link layer gets resolved for a next hop.

If the path selected by the device becomes unavailable, the IPv6 neighbor should change state and trigger the update of the destination path in the hardware.

Brocade FastIron devices support network-based ECMP load-sharing methods for IPv6 traffic. The Brocade device distributes traffic across equal-cost paths based on a XOR of some bits from the MAC source address, MAC destination address, IPv6 source address, IPv6 destination address, IPv6 flow label, IPv6 next header. The software selects a path based on a calculation involving the maximum number of load-sharing paths allowed and the actual number of paths to the destination network. This is the default ECMP load-sharing method for IPv6.

You can manually disable or enable ECMP load sharing for IPv6 and specify the number of equal-cost paths the device can distribute traffic across. In addition, you can display information about the status of ECMP load-sharing on the device.

Disabling or re-enabling ECMP load sharing for IPv6

ECMP load sharing for IPv6 is enabled by default. To disable the feature, enter the following command.

```
device(config)#no ipv6 load-sharing
```

If you want to re-enable the feature after disabling it, you must specify the number of load-sharing paths. By entering a command such as the following, IPv6 load-sharing will be re-enabled.

```
device(config)#ipv6 load-sharing 4
```

Syntax: [no] ipv6 load-sharing *num*

The *num* variable specifies the number of paths and can be from 2-8. The default is 4. On the ICX 7750 device, the value of the *num* variable can be from 2 through 32.

The configuration of the maximum number of IP load sharing paths to a value more than 8 is determined by the maximum number of ECMP paths defined at the system level using the **system-max max-ecmp** command. You cannot configure the maximum number of IP load sharing paths higher than the value defined at the system level.

To define the maximum number of ECMP paths at the system level, enter a command such as the following.

```
device(config)# system-max max-ecmp 20
device(config)# write memory
device(config)# exit
device# reload
```

Syntax: [no] system-max max-ecmp [*num*]

The *num* variable specifies the maximum number of ECMP paths and the value range can be from 8 through 32. This is supported only on the ICX 7750 device.

Changing the maximum load sharing paths for IPv6

By default, IPv6 ECMP load sharing allows traffic to be balanced across up to four equal paths.

To change the number of ECMP load sharing paths for IPv6, enter a command such as the following.

```
device(config)#ipv6 load-sharing 6
```

Syntax: [no] ipv6 load-sharing [*num*]

The *num* variable specifies the number of paths and can be from 2 through 8, depending on the device you are configuring. On the Brocade ICX 7750, the value of the *num* variable can be from 2 through 32.

The configuration of the maximum number of IP load sharing paths to a value more than 8 is determined by the maximum number of ECMP paths defined at the system level using the **system-max max-ecmp** command. You cannot configure the maximum number of IP load sharing paths higher than the value defined at the system level. Also, you cannot configure the maximum number of ECMP paths at the system level to a value less than the configured IP load sharing value.

To define the maximum number of ECMP paths at the system level, enter a command such as the following.

```
device(config)# system-max max-ecmp 20
device(config)# write memory
device(config)# exit
device# reload
```

Syntax: [no] system-max max-ecmp [*num*]

The *num* variable specifies the maximum number of ECMP paths and the value range can be from 8 through 32. This command is supported only on the Brocade ICX 7750.

You must save the configuration and reload the device for the maximum ECMP value change to take effect.

Displaying ECMP load-sharing information for IPv6

To display the status of ECMP load sharing for IPv6, enter the following command.

```
device#show ipv6
Global Settings
  unicast-routing enabled, hop-limit 64
  No IPv6 Domain Name Set
  No IPv6 DNS Server Address set
  Prefix-based IPv6 Load-sharing is Enabled, Number of load share paths: 4
```

Syntax: show ipv6

ICMP Router Discovery Protocol configuration

The ICMP Router Discovery Protocol (IRDP) is used by Brocade Layer 3 switches to advertise the IP addresses of its router interfaces to directly attached hosts. IRDP is disabled by default. You can enable the feature on a global basis or on an individual port basis:

- If you enable the feature globally, all ports use the default values for the IRDP parameters.
- If you leave the feature disabled globally but enable it on individual ports, you also can configure the IRDP parameters on an individual port basis.

NOTE

You can configure IRDP parameters only on an individual port basis. To do so, IRDP must be disabled globally and enabled only on individual ports. You cannot configure IRDP parameters if the feature is globally enabled.

When IRDP is enabled, the Layer 3 switch periodically sends Router Advertisement messages out the IP interfaces on which the feature is enabled. The messages advertise the Layer 3 switch IP addresses to directly attached hosts who listen for the messages. In addition, hosts can be configured to query the Layer 3 switch for the information by sending Router Solicitation messages.

Some types of hosts use the Router Solicitation messages to discover their default gateway. When IRDP is enabled on the Brocade Layer 3 switch, the Layer 3 switch responds to the Router Solicitation messages. Some clients interpret this response to mean that the Layer 3 switch is the default gateway. If another router is actually the default gateway for these clients, leave IRDP disabled on the Brocade Layer 3 switch.

IRDP parameters

IRDP uses the following parameters. If you enable IRDP on individual ports instead of enabling the feature globally, you can configure these parameters on an individual port basis:

- **Packet type** - The Layer 3 switch can send Router Advertisement messages as IP broadcasts or as IP multicasts addressed to IP multicast group 224.0.0.1. The packet type is IP broadcast.
- **Maximum message interval and minimum message interval** - When IRDP is enabled, the Layer 3 switch sends the Router Advertisement messages every 450 - 600 seconds by default. The time within this interval that the Layer 3 switch selects is random for each message and is not affected by traffic loads or other network factors. The random interval minimizes the probability that a host will receive Router Advertisement messages from other routers at the same time. The interval on each IRDP-enabled Layer 3 switch interface is independent of the interval on other IRDP-enabled interfaces. The default maximum message interval is 600 seconds. The default minimum message interval is 450 seconds.
- **Hold time** - Each Router Advertisement message contains a hold time value. This value specifies the maximum amount of time the host should consider an advertisement to be valid until a newer advertisement arrives. When a new advertisement arrives, the hold time is reset. The hold time is always longer than the maximum advertisement interval. Therefore, if the hold time for an

advertisement expires, the host can reasonably conclude that the router interface that sent the advertisement is no longer available. The default hold time is three times the maximum message interval.

- **Preference** - If a host receives multiple Router Advertisement messages from different routers, the host selects the router that sent the message with the highest preference as the default gateway. The preference can be a number from 0-4294967296. The default is 0.

Enabling IRDP globally

To globally enable IRDP, enter the following command.

```
device(config)# ip irdp
```

This command enables IRDP on the IP interfaces on all ports. Each port uses the default values for the IRDP parameters. The parameters are not configurable when IRDP is globally enabled.

Enabling IRDP on an individual port

To enable IRDP on an individual interface and change IRDP parameters, enter commands such as the following.

```
device(config)# interface ethernet 1/1/3
device(config-if-1/1/3)# ip irdp maxadvertinterval 400
```

This example shows how to enable IRDP on a specific port and change the maximum advertisement interval for Router Advertisement messages to 400 seconds.

NOTE

To enable IRDP on individual ports, you must leave the feature globally disabled.

Syntax: [no] ip irdp { **broadcast** | **multicast** } [**holdtime** *seconds*] [**maxadvertinterval** *seconds*] [**minadvertinterval** *seconds*] [**preference** *number*]

The **broadcast** and **multicast** parameters specify the packet type the Layer 3 switch uses to send Router Advertisement:

- **broadcast** - The Layer 3 switch sends Router Advertisement as IP broadcasts. This is the default.
- **multicast** - The Layer 3 switch sends Router Advertisement as multicast packets addressed to IP multicast group 224.0.0.1.

The **holdtime** *seconds* parameter specifies how long a host that receives a Router Advertisement from the Layer 3 switch should consider the advertisement to be valid. When a host receives a new Router Advertisement message from the Layer 3 switch, the host resets the hold time for the Layer 3 switch to the hold time specified in the new advertisement. If the hold time of an advertisement expires, the host discards the advertisement, concluding that the router interface that sent the advertisement is no longer available. The value must be greater than the value of the **maxadvertinterval** parameter and cannot be greater than 9000. The default is three times the value of the **maxadvertinterval** parameter.

The **maxadvertinterval** parameter specifies the maximum amount of time the Layer 3 switch waits between sending Router Advertisements. You can specify a value from 1 to the current value of the **holdtime** parameter. The default is 600 seconds.

The **minadvertinterval** parameter specifies the minimum amount of time the Layer 3 switch can wait between sending Router Advertisements. The default is three-fourths (0.75) the value of the **maxadvertinterval** parameter. If you change the **maxadvertinterval** parameter, the software automatically adjusts the **minadvertinterval** parameter to be three-fourths the new value of the **maxadvertinterval** parameter. If you want to override the automatically configured value, you can specify an interval from 1 to the current value of the **maxadvertinterval** parameter.

The **preference** *number* parameter specifies the IRDP preference level of this Layer 3 switch. If a host receives Router Advertisements from multiple routers, the host selects the router interface that sent the message with the highest interval as the host default gateway. The valid range is from 0 to 4294967296. The default is 0.

Configuring UDP broadcast and IP helper parameters

Some applications rely on client requests sent as limited IP broadcasts addressed to the UDP application port. If a server for the application receives such a broadcast, the server can reply to the client. Routers do not forward subnet directed broadcasts, so the client and server must be on the same network for the broadcast to reach the server. If the client and server are on different networks (on opposite sides of a router), the client request cannot reach the server.

You can configure the Layer 3 switch to forward clients' requests to UDP application servers. To do so:

- Enable forwarding support for the UDP application port, if forwarding support is not already enabled.
- Configure a helper address on the interface connected to the clients. Specify the helper address to be the IP address of the application server or the subnet directed broadcast address for the IP subnet the server is in. A helper address is associated with a specific interface and applies only to client requests received on that interface. The Layer 3 switch forwards client requests for any of the application ports the Layer 3 switch is enabled to forward to the helper address.

Forwarding support for the following application ports is enabled by default:

- dns (port 53)
- tftp (port 69)
- time (port 37)
- tacacs (port 65)

NOTE

The application names are the names for these applications that the Layer 3 switch software recognizes, and might not match the names for these applications on some third-party devices. The numbers listed in parentheses are the UDP port numbers for the applications. The numbers come from RFC 1340.

NOTE

Forwarding support for BootP/DHCP is enabled by default.

You can enable forwarding for other applications by specifying the application port number.

You also can disable forwarding for an application.

NOTE

If you disable forwarding for a UDP application, forwarding of client requests received as broadcasts to helper addresses is disabled. Disabling forwarding of an application does not disable other support for the application. For example, if you disable forwarding of Telnet requests to helper addresses, other Telnet support on the Layer 3 switch is not also disabled.

Enabling forwarding for a UDP application

If you want the Layer 3 switch to forward client requests for UDP applications that the Layer 3 switch does not forward by default, you can enable forwarding support for the port. To enable forwarding support for a UDP application, use the following method. You also can disable forwarding for an application using this method.

NOTE

You also must configure a helper address on the interface that is connected to the clients for the application. The Layer 3 switch cannot forward the requests unless you configure the helper address.

To enable the forwarding of NTP broadcasts, enter the following command.

```
device(config)# ip forward-protocol udp ntp
```

Syntax: [no] ip forward-protocol {udp *udp-port-name* | *udp-port-num* }

The *udp-port-name* parameter can have one of the following values. For reference, the corresponding port numbers from RFC 1340 are shown in parentheses. If you specify an application name, enter the name only, not the parentheses or the port number shown here:

- bootpc (port 68)
- bootps (port 67)
- discard (port 9)
- dns (port 53)
- dnsix (port 90)
- echo (port 7)
- mobile-ip (port 434)
- netbios-dgm (port 138)
- netbios-ns (port 137)
- ntp (port 123)
- tacacs (port 65)
- talk (port 517)
- time (port 37)
- tftp (port 69)

In addition, you can specify any UDP application by using the application UDP port number.

The *udp-port-num* parameter specifies the UDP application port number. If the application you want to enable is not listed above, enter the application port number. You also can list the port number for any of the applications listed above.

To disable forwarding for an application, enter a command such as the following.

```
device(config)# no ip forward-protocol udp ntp
```

This command disables forwarding of SNMP requests to the helper addresses configured on Layer 3 switch interfaces.

Configuring an IP helper address

To forward a client broadcast request for a UDP application when the client and server are on different networks, you must configure a helper address on the interface connected to the client. Specify the server IP address or the subnet directed broadcast address of the IP subnet the server is in as the helper address.

You can configure up to 16 helper addresses on each interface. You can configure a helper address on an Ethernet port or a virtual interface.

To configure a helper address on unit 1, slot 1, port 2, enter the following commands.

```
device(config)# interface ethernet 1/1/2
device(config-if-1/1/2)# ip helper-address 1 10.95.7.6
```

The commands in this example change the CLI to the configuration level for port 1/1/2, then add a helper address for server 10.95.7.6 to the port. If the port receives a client request for any of the applications that the Layer 3 switch is enabled to forward, the Layer 3 switch forwards the client request to the server.

By default, IP helper does not forward client broadcast request to a server within the network.

To forward a client broadcast request when the client and server are on the same network, configure an IP helper with unicast option on the interface connected to the client.

To configure an IP helper unicast option on unit 1, slot 1, port 2, enter the following commands:

```
device(config)# interface 1/1/2
device(config-if-1/1/2)# ip helper-address 1 10.10.10.1 unicast
```

The IP helper with unicast parameter forwards the client request to the server 10.10.10.1 which is within the network.

Syntax: `ip helper-address num ip-addr [unicast]`

The *num* variable specifies the helper address number and can be from 1 through 16.

The *ip-addr* variable specifies the server IP address or the subnet directed broadcast address of the IP subnet the server is in.

The **unicast** parameter specifies that the client request must be forwarded to the server that is on the same network.

Configuring IP parameters - Layer 2 switches

The following sections describe how to configure IP parameters on a Brocade Layer 2 switch. To manage a Layer 2 switch using Telnet or Secure Shell (SSH) CLI connections or the Web Management Interface, you must configure an IP address for the Layer 2 switch.

Optionally, you also can specify the default gateway.

Configuring the management IP address and specifying the default gateway

Brocade devices support both classical IP network masks (Class A, B, and C subnet masks, and so on) and Classless Interdomain Routing (CIDR) network prefix masks:

- To enter a classical network mask, enter the mask in IP address format. For example, enter "10.157.22.99 255.255.255.0" for an IP address with a Class-C subnet mask.
- To enter a prefix network mask, enter a forward slash (/) and the number of bits in the mask immediately after the IP address. For example, enter "10.157.22.99/24" for an IP address that has a network mask with 24 significant bits (ones).

By default, the CLI displays network masks in classical IP address format (example: 255.255.255.0). You can change the display to prefix format.

Assigning an IP address to a Brocade Layer 2 switch

To assign an IP address to a Brocade Layer 2 switch, enter a command such as the following at the global CONFIG level.

```
device(config)# ip address 10.45.6.110 255.255.255.0
```

Syntax: `ip address ip-addr ip-mask`

or

Syntax: `ip address ip-addr/mask-bits`

You also can enter the IP address and mask in CIDR format, as follows.

```
device(config)# ip address 10.45.6.1/24
```

To specify the Layer 2 switch default gateway, enter a command such as the following.

```
device(config)# ip default-gateway 10.45.6.1
```

Syntax: `ip default-gateway ip-addr`

NOTE

When configuring an IP address on a Layer 2 switch that has multiple VLANs, make sure the configuration includes a designated management VLAN that identifies the VLAN to which the global IP address belongs. Refer to "Designated VLAN for Telnet management sessions to a Layer 2 Switch" in the *Brocade FastIron Security Configuration Guide*.

Configuring Domain Name System resolver

The Domain Name System (DNS) resolver feature lets you use a host name to perform Telnet, ping, and traceroute commands. You can also define a DNS domain on a Brocade Layer 2 switch or Layer 3 switch and thereby recognize all hosts within that domain. After you define a domain name, the Brocade Layer 2 switch or Layer 3 switch automatically appends the appropriate domain to the host and forwards it to the domain name server.

For example, if the domain "newyork.com" is defined on a Brocade Layer 2 switch or Layer 3 switch and you want to initiate a ping to host "NYC01" on that domain, you need to reference only the host name in the command instead of the host name and its domain name. For example, you could enter either of the following commands to initiate the ping.

```
device# ping nyc01
device# ping nyc01.newyork.com
```

Defining a DNS entry

You can define up to four DNS servers for each DNS entry. The first entry serves as the primary default address. If a query to the primary address fails to be resolved after three attempts, the next gateway address is queried (also up to three times). This process continues for each defined gateway address until the query is resolved. The order in which the default gateway addresses are polled is the same as the order in which you enter them.

To define four possible default DNS gateway addresses, enter command such as the following:

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

Syntax: `ip dns server-address ip-addr [ip-addr] [ip-addr] [ip-addr]`

In this example, the first IP address in the `ip dns server-address` command becomes the primary gateway address and all others are secondary addresses. Because IP address 10.98.7.15 is the last address listed, it is also the last address consulted to resolve a query.

Using a DNS name to initiate a trace route

Suppose you want to trace the route from a Brocade Layer 2 switch to a remote server identified as NYC02 on domain newyork.com. Because the newyork.com domain is already defined on the Layer 2 switch, you need to enter only the host name, NYC02, as noted in the following command.

```
device# traceroute nyc02
```

Syntax: `traceroute host-ip-addr [maxttl value] [minttl value] [numeric] [timeout value] [source-ip ip-addr]`

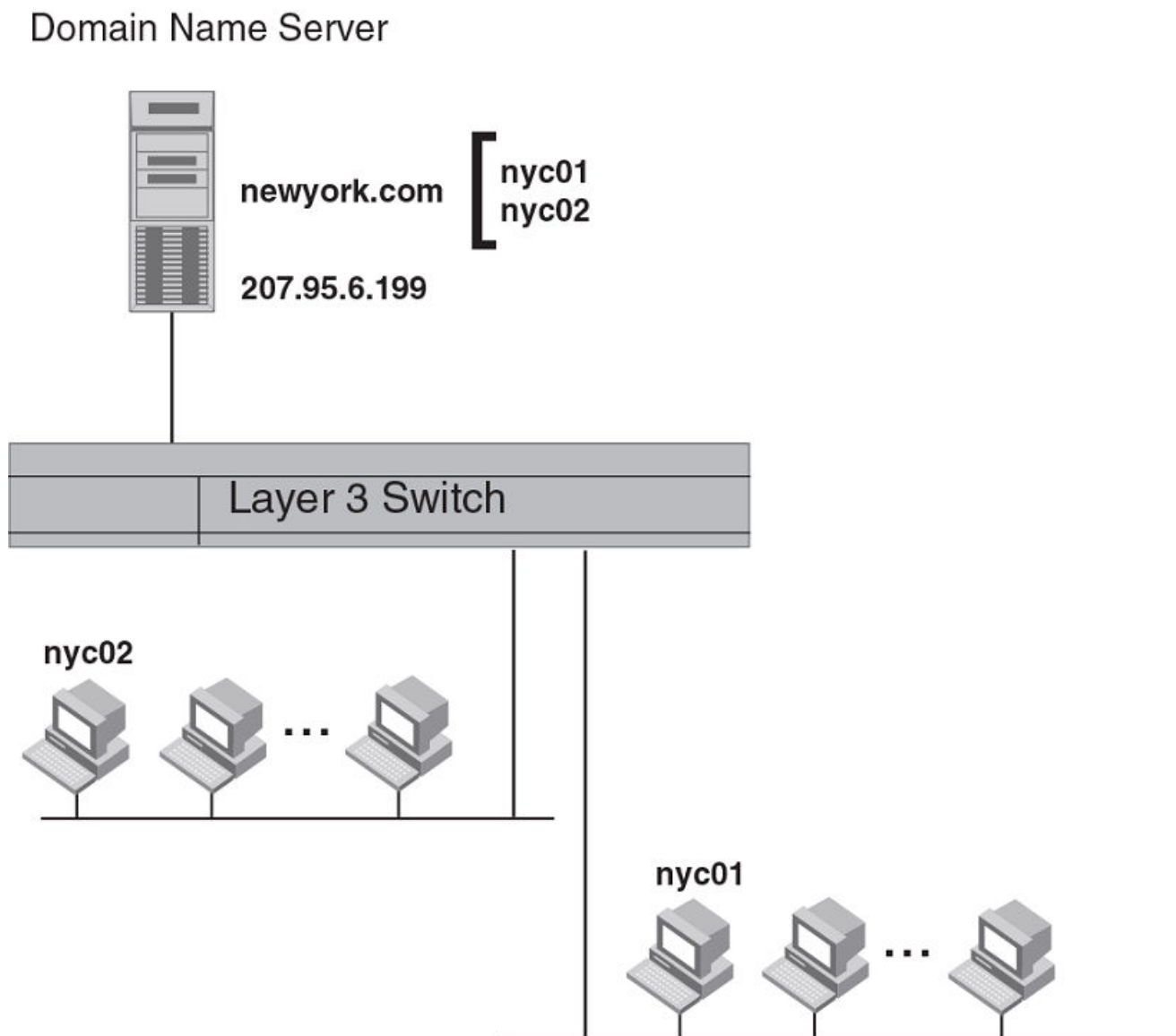
The only required parameter is the IP address of the host at the other end of the route.

After you enter the command, a message indicating that the DNS query is in process and the current gateway address (IP address of the domain name server) being queried appear on the screen.

```
Type Control-c to abort
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
```

NOTE

In the previous example, 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 NYC02 host.

FIGURE 6 Querying a host on the newyork.com domain

Changing the TTL threshold

The time to live (TTL) threshold prevents routing loops by specifying the maximum number of router hops an IP packet originated by the Layer 2 switch can travel through. Each device capable of forwarding IP that receives the packet decrements (decreases) the packet TTL by one. If a router receives a packet with a TTL of 1 and reduces the TTL to zero, the router drops the packet.

The default TTL is 64. You can change the *ttl-threshold* to a value from 1 through 255.

To modify the TTL threshold to 25, enter the following commands.

```
device(config)# ip ttl 25
device(config)# exit
```

Syntax: `ip ttl ttl-threshold`

IPv4 point-to-point GRE tunnels

This section describes support for point-to-point Generic Routing Encapsulation (GRE) tunnels and how to configure them on a Brocade device.

GRE tunnels support includes the following:

- IPv4 over GRE tunnels. IPv6 over GRE tunnels is not supported.
- Static and dynamic unicast routing over GRE tunnels
- Multicast routing over GRE tunnels
- Hardware forwarding of IP data traffic across a GRE tunnel.
- Path MTU Discovery (PMTUD)

IPv4 GRE tunnel overview

Generic Routing Encapsulation is described in RFC 2784. Generally, GRE provides a way to encapsulate arbitrary packets (payload packet) inside of a transport protocol, and transmit them from one tunnel endpoint to another. The payload is encapsulated in a GRE packet. The resulting GRE packet is then encapsulated in a delivery protocol, then forwarded to the tunnel destination. At the tunnel destination, the packet is decapsulated to reveal the payload. The payload is then forwarded to its final destination.

Brocade devices allow the tunneling of packets of the following protocols over an IPv4 network using GRE:

- OSPF V2
- BGP4
- RIP V1 and V2

GRE packet structure and header format

FIGURE 7 GRE encapsulated packet structure

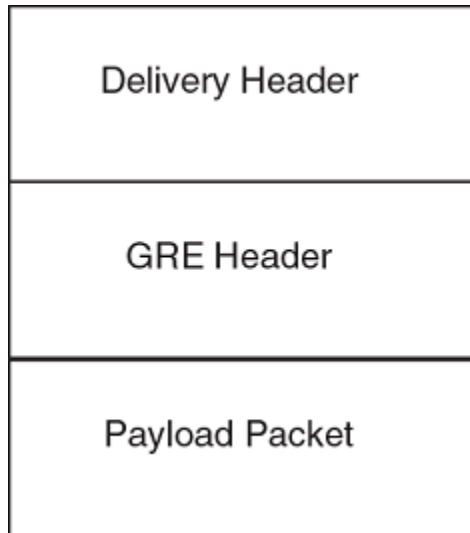


FIGURE 8 GRE header format

Checksum	Reserved0	Ver	Protocol Type	Checksum (optional)	Reserved (optional)
----------	-----------	-----	---------------	---------------------	---------------------

The GRE header has the following fields:

- Checksum - 1 bit. This field is assumed to be zero in this version. If set to 1, this means that the Checksum **(optional)** and Reserved **(optional)** fields are present and the **Checksum (optional)** field contains valid information.
- Reserved0 - 12 bits. If bits 1 - 5 are non-zero, then a receiver must discard the packet unless RFC 1701 is implemented. Bits 6 - 12 are reserved for future use and must be set to zero in transmitted packets. This field is assumed to be zero in this version.
- Ver - 3 bits. The GRE protocol version. This field must be set to zero in this version.
- Protocol Type - 16 bits. The Ethernet protocol type of the packet, as defined in RFC 1700.
- Checksum (optional) - 16 bits. This field is optional. It contains the IP checksum of the GRE header and the payload packet.
- Reserved (optional) - 16 bits. This field is optional. It is reserved for Brocade internal use.

Path MTU Discovery support

Brocade IronWare software supports the following RFCs for handling large packets over a GRE tunnel:

- RFC 1191, Path MTU Discovery
- RFC 4459, MTU and Fragmentation Issues with In-the-Network Tunneling

RFC 1191 describes a method for dynamically discovering the maximum transmission unit (MTU) of an arbitrary internet path. When a FastIron device receives an IP packet that has its Do not Fragment (DF) bit set, and the packet size is greater than the MTU value of the outbound interface, then the FastIron device returns an ICMP Destination Unreachable message to the source of the packet, with the code indicating "fragmentation needed and DF set". The ICMP Destination Unreachable message includes the MTU of the outbound interface. The source host can use this information to help determine the minimum MTU of a path to a destination.

RFC 4459 describes solutions for issues with large packets over a tunnel. The following methods, from RFC 4459, are supported in Brocade IronWare software:

- If a source attempts to send packets that are larger than the lowest MTU value along the path, Path MTU Discovery (PMTUD) can signal to the source to send smaller packets. This method is described in Section 3.2 of RFC 4459.
- Inner packets can be fragmented before encapsulation, in such a manner that the encapsulated packet fits in the tunnel path MTU, which is discovered using PMTUD. This method is described in Section 3.4 of RFC 4459.

By default, PMTUD is enabled.

Support for IPv4 multicast routing over GRE tunnels

PIM-DM and PIM-SM Layer 3 multicast protocols and multicast data traffic are supported over GRE tunnels. When a multicast protocol is enabled on both ends of a GRE tunnel, multicast packets can be sent from one tunnel endpoint to another. To accomplish this, the packets are encapsulated using the GRE unicast tunneling mechanism and forwarded like any other IPv4 unicast packet to the destination endpoint of the tunnel. The router that terminates the tunnel (i.e., the router where the tunnel endpoint is an ingress interface) de-encapsulates the GRE tunneled packet to retrieve the native multicast data packets. After de-encapsulation, data packets are forwarded in the direction of its receivers, and control packets may be consumed. This creates a PIM-enabled virtual or logical link between the two GRE tunnel endpoints.

Strict RPF check for multicast protocols

Brocade software enforces strict Reverse Path Forwarding (RPF) check rules on an (s,g) entry on a GRE tunnel interface. The (s,g) entry uses the GRE tunnel as an RPF interface. During unicast routing transit, GRE tunnel packets may arrive at different physical interfaces. The strict RPF check limits GRE PIM tunnel interfaces to accept the (s,g) GRE tunnel traffic.

Configuration considerations for GRE IP tunnels

Before configuring GRE tunnels and tunnel options, consider the configuration notes in this section.

- When GRE is enabled on a Layer 3 switch, the following features are not supported on Virtual Ethernet (VE) ports, VE member ports (ports that have IP addresses), and GRE tunnel loopback ports:
 - ACL logging
 - ACL statistics (also called ACL counting)
 - MAC address filters
 - IPv6 filters

NOTE

The above features are supported on VLANs that do not have VE ports.

- Whenever multiple IP addresses are configured on a tunnel source, the primary address of the tunnel is always used for forming the tunnel connections. Therefore, carefully check the configurations when configuring the tunnel destination.
- When a GRE tunnel is configured, you cannot configure the same routing protocol on the tunnel through which you learn the route to the tunnel destination. For example, if the FastIron learns the tunnel destination route through the OSPF protocol, you

cannot configure the OSPF protocol on the same tunnel and vice-versa. When a tunnel has OSPF configured, the FastIron cannot learn the tunnel destination route through OSPF. This could cause the system to become unstable.

- The tunnel destination cannot be resolved to the tunnel itself or any other local tunnel. This is called recursive routing. This scenario would cause the tunnel interface to flap and the Syslog message TUN-RECURSIVE-DOWN to be logged. To resolve this issue, create a static route for the tunnel destination.

GRE MTU configuration considerations

When jumbo is enabled, the default Ethernet MTU size is 9216 bytes. The maximum Ethernet MTU size is 10218 bytes. The MTU of the GRE tunnel is compared with the outgoing packet before the packet is encapsulated. After encapsulation, the packet size increases by 24 bytes. Therefore, when changing the GRE tunnel MTU, set the MTU to at least 24 bytes less than the IP MTU of the outgoing interface. If the MTU is not set to at least 24 bytes less than the IP MTU, the size of the encapsulated packet will exceed the IP MTU of the outgoing interface. This will cause the packet to either be sent to the CPU for fragmentation, or the packet will be dropped if the DF (Do-Not-Fragment) bit is set in the original IP packet, and an ICMP message is sent.

NOTE

The fragmentation behavior depends on the mtu-exceed setting on the router.

Configuration tasks for GRE tunnels

Perform the configuration tasks in the order listed.

TABLE 9 Configuration tasks for GRE tunnels

Configuration tasks	Default behavior
Required tasks	
Create a tunnel interface.	Not assigned
Configure the source address or source interface for the tunnel interface.	Not assigned
Configure the destination address of the tunnel interface.	Not assigned
Enable GRE encapsulation on the tunnel interface.	Disabled
NOTE Step 4 must be performed before step 6.	
If packets need to be terminated in hardware, configure a tunnel loopback port for the tunnel interface. NOTE Step 5 is not applicable to FCX devices.	Not assigned
Configure an IP address for the tunnel interface.	Not assigned
If a route to the tunnel destination does not already exist, create a static route and specify that the route is through the tunnel interface.	Not assigned
Optional tasks	
Change the maximum transmission unit (MTU) value for the tunnel interface.	1476 bytes or 9192 bytes (jumbo mode)
Change the number of GRE tunnels supported on the device.	Support for 32 GRE tunnels
Enable and configure GRE link keepalive on the tunnel interface.	Disabled
Change the Path MTU Discovery (PMTUD) configuration on the GRE tunnel interface.	Enabled
Enable support for IPv4 multicast routing.	Disabled

The following features are also supported on GRE tunnel interfaces:

- Naming the tunnel interface (CLI command **port-name**).
- Changing the Maximum Transmission Unit (MTU) (CLI command **ip mtu**).
- Increasing the cost of routes learned on the port (CLI command **ip metric**).

After configuring GRE tunnels, you can view the GRE configuration and observe the routes that use GRE tunnels.

Creating a tunnel interface

To create a tunnel interface, enter the following command at the Global CONFIG level of the CLI.

```
device(config)# interface tunnel 1
device(config-tnif-1)#
```

Syntax: [no] **interface tunnel** *tunnel-number*

The *tunnel-number* is a numerical value that identifies the tunnel being configured.

NOTE

You can also use the **port-name** command to name the tunnel. To do so, follow the configuration instructions in "Assigning a port name" section in the *Brocade FastIron Management Configuration Guide*.

Assigning a VRF routing instance to a GRE tunnel interface

A GRE tunnel interface can be assigned to an existing user defined VRF. When the VRF is configured on a tunnel, all IPv4 and IPv6 addresses are removed. The tunnel loopback configuration is removed.

To assign the VRF named VRF1 to tunnel 1, enter the following commands.

```
Brocade(config)# interface tunnel 1
Brocade(config-tnif-1)# vrf forwarding VRF1
```

Syntax: [no] **vrf forwarding** *vrf-name*

The *vrf-name* variable is the name of the VRF that the interface is being assigned to.

Configuring the source address or source interface for a tunnel interface

To configure the source for a tunnel interface, specify either a source address or a source interface.

NOTE

If the destination address for a tunnel interface is not resolved, Brocade recommends that you either configure the *source interface* (instead of the *source address*) as the source for a tunnel interface, or enable GRE link keepalive on the tunnel interface.

The tunnel source address should be one of the router IP addresses configured on a physical, loopback, or VE interface, through which the other end of the tunnel is reachable.

To configure the source address for a specific tunnel interface, enter commands such as the following.

```
device(config)# interface tunnel 1
device(config-tnif-1)# tunnel source 10.0.8.108
```

The source interface should be the port number of the interface configured on a physical, loopback, or VE interface. The source interface should have at least one IP address configured on it. Otherwise, the interface will not be added to the tunnel configuration and an error message similar to the following will be displayed:

```
ERROR - Tunnel source interface 1/3/1 has no configured IP address.
```

To configure the source interface for a specific tunnel interface, enter commands such as the following.

```
device(config)# interface tunnel 1
device(config-tnif-1)# tunnel source ethernet 1/3/1
```

Syntax: [no] **tunnel source** { *ip-address* | **ethernet** *unit / slot / port* | **ve** *number* | **loopback** *number* }

The *ip-address* variable is the source IP address being configured for the specified tunnel.

The **ethernet** *unit / slot / port* parameter identifies a physical interface being configured for the specified tunnel, for example 1/3/1.

The *ve number* variable is the VE interface number being configured for the specified tunnel.

Deleting an IP address from an interface configured as a tunnel source

To delete an IP address from an interface that is configured as a tunnel source, first remove the tunnel source from the tunnel interface then delete the IP address, as shown in the following example.

```
device(config-if-e1000-1/1/3)# interface tunnel 8
device(config-tnif-8)# no tunnel source 10.1.83.15
device(config-tnif-8)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# no ip address 10.1.83.15/24
```

If you attempt to delete an IP address without first removing the tunnel source, the console will display an error message, as shown in the following example.

```
device# config terminal
device(config)# interface ethernet 1/1/3
device(config-if-e1000-1/1/3)# no ip address 10.1.83.15/24
Error - Please remove tunnel source from tnnl 8 before removing IP address
```

NOTE

The previous error message will also display on the CLI when an interface is part of a VLAN. A VLAN cannot be deleted until the tunnel source is first removed.

Configuring the destination address for a tunnel interface

The destination address should be the address of the IP interface of the device on the other end of the tunnel.

To configure the destination address for a specific tunnel interface, enter commands such as the following.

```
device(config)# interface tunnel 1
device(config-tnif-1)# tunnel destination 131.108.5.2
```

Syntax: [no] **tunnel destination** *ip-address*

The *ip-address* variable is the destination IP address being configured for the specified tunnel.

NOTE

Ensure a route to the tunnel destination exists on the tunnel source device. Create a static route if necessary.

Enabling GRE encapsulation on a tunnel interface

To enable GRE encapsulation on a tunnel interface, enter commands such as the following.

```
device(config)# interface tunnel 1
device(config-tnif-1)# tunnel mode gre ip
```

Syntax: [no] tunnel mode gre ip

- **gre** specifies that the tunnel will use GRE encapsulation (IP protocol 47).
- **ip** specifies that the tunneling protocol is IPv4.

NOTE

Before configuring a new GRE tunnel, the system should have at least one slot available for adding the default tunnel MTU value to the system tables. Depending on the configuration, the default tunnel MTU range is ((1500 or 10218) - 24) . To check for slot availability, or to see if the MTU value is already configured in the IP table, use the **show ip mtu** command.

Configuring a tunnel loopback port for a tunnel interface

For details and important configuration considerations regarding tunnel loopback ports for GRE tunnels, refer to the “Tunnel loopback ports for GRE tunnels” task and the “Configuration considerations for tunnel loopback ports” task.

To configure a tunnel loopback port, enter commands such as the following:

```
device(config)# interface tunnel 1
device(config-tnif-1)# tunnel loopback 1/3/1
```

Syntax: [no] tunnel loopback unit / slot / port

The *unit / slot / port* parameter identifies the tunnel loopback port for the specified tunnel interface, for example, 1/3/1.

Configuring an IP address for a tunnel interface

An IP address sets a tunnel interface as an IP port and allows the configuration of Layer 3 protocols, such as OSPF, BGP, and Multicast (PIM-DM and PIM-SM) on the port. Note that the subnet cannot overlap other subnets configured on other routing interfaces, and both ends of the tunnel should be in the same subnet.

To configure an IP address for a specified tunnel interface, enter commands such as the following.

```
device(config)# interface tunnel 1
device(config-tnif-1)# ip address 10.10.3.1/24
```

Syntax: [no] ip address ip-address

The *ip-address* is the IP address being configured for the specified tunnel interface.

Configuring a static route to a tunnel destination

If a route to the tunnel destination does not already exist on the tunnel source, create a static route and set the route to go through the tunnel interface.

```
device(config)# ip route 131.108.5.0/24 10.0.8.1
device(config)# ip route 10.10.2.0/24 tunnel 1
```

Syntax: [no] ip route ip-address tunnel tunnel-ID

- The *ip-address* variable is the IP address of the tunnel interface.
- The *tunnel-ID* variable is a valid tunnel number or name.

Changing the MTU value for a tunnel interface

For important configuration considerations regarding this feature, refer to [GRE MTU configuration considerations](#) on page 88.

You can set an MTU value for packets entering the tunnel. Packets that exceed either the default MTU value of 1476/9192 bytes (for jumbo case) or the value that you set using this command, are fragmented and encapsulated with IP/GRE headers for transit through the tunnel (if they do not have the DF bit set in the IP header). All fragments will carry the same DF bit as the incoming packet. Jumbo packets are supported, although they may be fragmented based on the configured MTU value.

The following command allows you to change the MTU value for packets transiting "tunnel 1":

```
device(config)# interface tunnel 1
device(config-tnif-1)# ip mtu 1200
```

Syntax: ip mtu *packet-size*

The *packet-size* variable specifies the maximum size in bytes for the packets transiting the tunnel. Enter a value from 576 through 1476. The default value is 1476.

NOTE

To prevent packet loss after the 24 byte GRE header is added, make sure that any physical interface that is carrying GRE tunnel traffic has an IP MTU setting at least 24 bytes greater than the tunnel MTU setting. This configuration is only allowed on the system if the tunnel mode is set to GRE.

Changing the maximum number of tunnels supported

Use the following table to determine how many GRE tunnels are supported. You can configure the device to support up to the maximum number of GRE tunnels as displayed in the following table.

Device	Max # of GRE tunnels	Default # of GRE tunnels
ICX 7250	8	8
ICX 7420	64	16
ICX 7750	64	16

To change the maximum number of tunnels supported, enter commands such as the following.

```
device(config)# system-max gre-tunnels 16
Reload required. Please write memory and then reload or power cycle.
device(config)# write memory
device(config)# exit
device# reload
```

NOTE

You must save the configuration (write memory) and reload the software to place the change into effect.

Syntax: system-max gre-tunnels *number*

The *number* variable specifies the number of GRE tunnels that can be supported on the device. The permissible range is 16 - 64. The **system-max gre-tunnels** command determines the interface range that is supported for an interface tunnel. For example, if the system-max value is reduced, it is possible that the configured interfaces may be rejected after a system reload.

Configuring GRE link keepalive

When GRE tunnels are used in combination with static routing or policy-based routing, and a dynamic routing protocol such as RIP, BGP, or OSPF is not deployed over the GRE tunnel, a configured tunnel does not have the ability to bring down the line protocol of either tunnel endpoint, if the far end becomes unreachable. Traffic sent on the tunnel cannot follow alternate paths because the tunnel is always UP. To avoid this scenario, enable GRE link keepalive, which will maintain or place the tunnel in an UP or DOWN state based upon the periodic sending of keepalive packets and the monitoring of responses to the packets. If the packets fail to reach the tunnel far end more frequently than the configured number of retries, the tunnel is placed in the DOWN state.

To enable GRE link keepalive, configure it on one end of the tunnel and ensure the other end of the tunnel has GRE enabled.

NOTE

Keepalives are not supported when a tunnel interface is not within the default-VRF.

To configure GRE link keepalive, enter commands such as the following.

```
device(config)# interface tunnel 1
device(config-tnif-1)# keepalive 12 4
```

These commands configure the device to wait for 4 consecutive lost keepalive packets before bringing the tunnel down. There will be a 12 second interval between each packet. Note that when the tunnel comes up, it would immediately (within one second) send the first keepalive packet.

Syntax: **[no] keepalive** seconds *retries*

Use the **no** form of the command to disable the keepalive option.

The *seconds* variable specifies the number of seconds between each initiation of a keepalive message. The range for this interval is 2 - 32767 seconds. The default value is 10 seconds.

The *retries* variable specifies the number of times that a packet is sent before the system places the tunnel in the DOWN state. Possible values are from 1 through 255. The default number of retries is 3.

Use the **show interface tunnel** and **show ip tunnel traffic** commands to view the GRE link keepalive configuration.

Configuring Path MTU Discovery (PMTUD)

PMTUD is enabled by default on tunnel interfaces. This section describes how to disable and re-enable PMTUD on a tunnel interface, change the PMTUD age timer, manually clear the tunnel PMTUD, and view the PMTUD configuration.

Disabling and re-enabling PMTUD

PMTUD is enabled by default. To disable it, enter the following command:

```
device(config-tnif-1)# tunnel path-mtu-discovery disable
```

To re-enable PMTUD after it has been disabled, enter the following command:

```
device(config-tnif-1)# no tunnel path-mtu-discovery disable
```

Syntax: **[no] tunnel path-mtu-discovery disable**

Changing the age timer for PMTUD

By default, when PMTUD is enabled on a tunnel interface, the path MTU is reset to its original value every 10 minutes. If desired, you can change the reset time (default age timer) to a value of up to 30 minutes. To do so, enter a command such as the following on the GRE tunnel interface.

```
device(config-tnif-1) # tunnel path-mtu-discovery age-timer 20
```

This command configures the device to wait for 20 minutes before resetting the path MTU to its original value.

Syntax: `[no] tunnel path-mtu-discovery { age-timer minutes | infinite }`

For *minutes*, enter a value from 10 to 30.

Enter **infinite** to disable the timer.

Clearing the PMTUD dynamic value

To reset a dynamically-configured MTU on a tunnel Interface back to the configured value, enter the following command.

```
device(config) # clear ip tunnel pmtud 1
```

Syntax: `clear ip tunnel pmtud tunnel-ID`

The *tunnel-ID* variable is a valid tunnel number or name.

Viewing PMTUD configuration details

Use the **show interface tunnel** command to view the PMTUD configuration and to determine whether PMTUD has reduced the size of the MTU.

Enabling IPv4 multicast routing over a GRE tunnel

This section describes how to enable IPv4 multicast protocols, PIM Sparse (PIM-SM) and PIM Dense (PIM-DM), on a GRE tunnel.

Perform the procedures in this section after completing the required tasks in [Enabling IPv4 multicast routing over a GRE tunnel](#).

For an overview of multicast routing support over a GRE tunnel, refer to [Support for IPv4 multicast routing over GRE tunnels](#) on page 87. To view information about multicast protocols and GRE tunnel-specific information, refer to [Displaying multicast protocols and GRE tunneling information](#) on page 98.

Enabling PIM-SM on a GRE tunnel

To enable PIM-SM on a GRE tunnel interface, enter commands such as the following:

```
device(config) # interface tunnel 10
device(config-tnif-10) # ip pim-sparse
```

Syntax: `[no] ip pim-sparse`

Use the **no** form of the command to disable PIM-SM on the tunnel interface.

Enabling PIM-DM on a GRE tunnel interface

To enable PIM-DM on a GRE tunnel interface, enter commands such as the following:

```
device(config) # interface tunnel 10
device(config-tnif-10) # ip pim
```

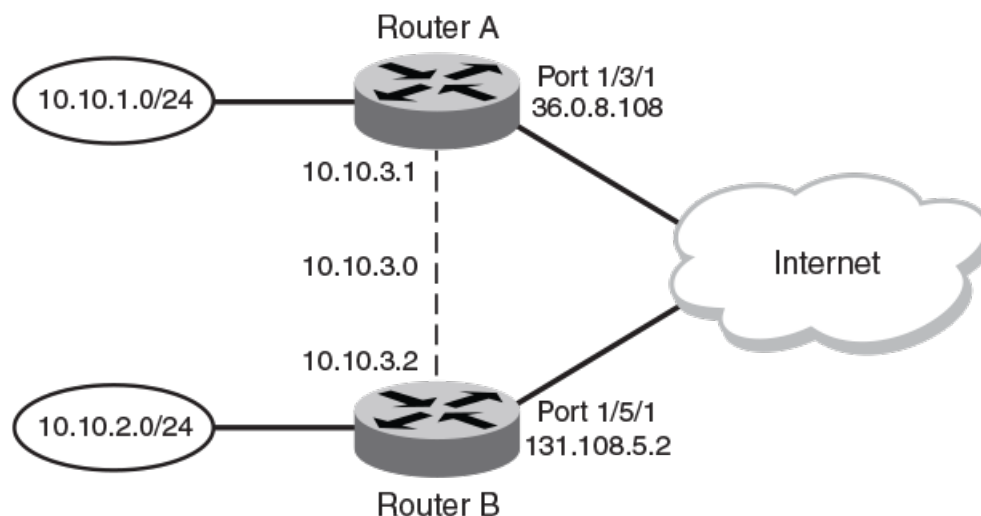
Syntax: [no] ip pim

Use the **no** form of the command to disable PIM-DM on the tunnel interface.

Example point-to-point GRE tunnel configuration

A GRE Tunnel is configured between Router A and Router B. Traffic between networks 10.10.1.0/24 and 10.10.2.0/24 is encapsulated in a GRE packet sent through the tunnel on the 10.10.3.0 network, and unpacked and sent to the destination network. A static route is configured at each Layer 3 switch to go through the tunnel interface to the target network.

FIGURE 9 Point-to-point GRE tunnel configuration example



The following shows the configuration commands for this example.

Configuring point-to-point GRE tunnel for Router A

```
device (config)# interface ethernet 1/3/1
device (config-if-e1000-1/3/1)# ip address 10.0.8.108/24
device (config)# exit
device (config)# interface tunnel 1
device (config-tnif-1)# tunnel source 10.0.8.108
device (config-tnif-1)# tunnel destination 131.108.5.2
device (config-tnif-1)# tunnel mode gre ip
device (config-tnif-1)# tunnel loopback 4/1
device (config-tnif-1)# ip address 10.10.3.1/24
device (config-tnif-1)# exit
device (config)# ip route 131.108.5.0/24 10.0.8.1
device (config)# ip route 10.10.2.0/24 tunnel 1
```

Configuring point-to-point GRE tunnel for Router B

```
device (config)# interface ethernet 1/5/1
device (config-if-e1000-1/5/1)# ip address 131.108.5.2/24
device (config)# exit
device (config)# interface tunnel 1
device (config-tnif-1)# tunnel source 131.108.5.2
device (config-tnif-1)# tunnel destination 10.0.8.108
device (config-tnif-1)# tunnel mode gre ip
device (config-tnif-1)# tunnel loopback 1/1
```

```

device(config-tnif-1)# ip address 10.10.3.2/24
device(config-tnif-1)# exit
device(config)# ip route 10.0.8.0/24 131.108.5.1
device(config)# ip route 10.10.1.0/24 tunnel

```

Displaying GRE tunneling information

This section describes the **show** commands that display the GRE tunnels configuration, the link status of the GRE tunnels, and the routes that use GRE tunnels.

To display GRE tunneling Information, use the following commands:

- **show ip interface**
- **show ip route**
- **show ip interface tunnel**
- **show ip tunnel traffic**
- **show interface tunnel**
- **show statistics tunnel**

The following shows an example output of the **show ip interface** command, which includes information about GRE tunnels.

```

device# show ip interface
Interface  IP-Address  OK?  Method  Status  Protocol  VRF
Tunnel 1   101.1.1.1    YES  NVRAM   up       up         red
Tunnel 3   89.1.1.1     YES  NVRAM   up       up         default-vrf

```

For field definitions, refer to the *FastIron Command Reference*.

Syntax: show ip interface

The **show ip route** command displays routes that are pointing to a GRE tunnel as shown in the following example.

```

device# show ip route
Total number of IP routes: 3, avail: 79996 (out of max 80000)
B:BGP D:Connected R:RIP S:Static O:OSPF *:Candidate default
Destination      NetMask          Gateway          Port          Cost    Type
1    10.1.1.0        255.255.255.0    0.0.0.0        7          1      D
2    10.1.2.0        255.255.255.0    10.1.1.3       7          1      S
3    10.34.3.0       255.255.255.0    0.0.0.0        tn3          1      D

```

For field definitions, refer to *FastIron Command Reference*.

Syntax: show ip route

The **show ip interface tunnel** command displays the link status and IP address configuration for an IP tunnel interface as shown in the following example.

```

device# show ip interface tunnel 64
Interface Tunnel 64
port enabled
port state: UP
ip address: 223.224.64.0/31
Port belongs to VRF: default-vrf
encapsulation: GRE, mtu: 1476, metric: 1
directed-broadcast-forwarding: disabled
proxy-arp: disabled
ip arp-age: 10 minutes
No Helper Addresses are configured.
No inbound ip access-list is set
No outgoing ip access-list is set

```

Syntax: show ip interface tunnel [tunnel-ID]

The *tunnel-ID* variable is a valid tunnel number between 1 and 72.

The **show interface tunnel** command displays the GRE tunnel configuration and the pmtd aging timer information.

```
device# show interface tunnel 10
Tunnel10 is up, line protocol is up
  Hardware is Tunnel
  Tunnel source 1.1.41.10
  Tunnel destination is 1.1.14.10
  Tunnel mode gre ip
  Port name is GRE_10_to_VR1_on_ICX_STACK
  Internet address is 223.223.1.1/31, MTU 1476 bytes, encapsulation GRE
  Keepalive is not Enabled
  Path MTU Discovery: Enabled, MTU is 1428 bytes, age-timer: 10 minutes
  Path MTU will expire in 0 minutes 50 secs
```

Syntax: **show interface tunnel** [*tunnel-ID*]

TABLE 10 show interface tunnel output descriptions

Field	Definition
Hardware is Tunnel	The interface is a tunnel interface.
Tunnel source	The source address for the tunnel.
Tunnel destination	The destination address for the tunnel.
Tunnel mode	The tunnel mode. The gre specifies that the tunnel will use GRE encapsulation (IP protocol 47).
Port name	The port name (if applicable).
Internet address	The internet address.
MTU	The configured path maximum transmission unit.
encapsulation GRE	GRE encapsulation is enabled on the port.
Keepalive	Indicates whether or not GRE link keepalive is enabled.
Path MTU Discovery	Indicates whether or not PMTUD is enabled. If PMTUD is enabled, the MTU value is also displayed.
Path MTU	The PMTU that is dynamically learned.
Age-timer	Indicates the pmtd aging timer configuration in minutes. The default is 10. The range is from 10 - 30.
Path MTU will expire	Indicates the time after which the learned PMTU expires. This line is displayed only when a PMTU is dynamically learned.

The **show ip tunnel traffic** command displays the link status of the tunnel and the number of keepalive packets received and sent on the tunnel.

```
device# show ip tunnel traffic
IP GRE Tunnels
  Tunnel Status  Packet Received  Packet Sent  KA recv  KA sent
1    up/up        362              0           362      362
3    up/up        0                0           0        0
10   down/down    0                0           0        0
```

Syntax: **show ip tunnel traffic**

The **show statistics tunnel** command displays GRE tunnel statistics for a specific tunnel ID number. The following shows an example output for tunnel ID 1.

```
device(config-tnif-10)# show statistics tunnel 1
IP GRE Tunnels
  Tunnel Status  Packet Received  Packet Sent  KA recv  KA sent
1    up/up        87120           43943      43208    43855
```

RFC 2784 supports GRE tunnel ports. The show statistics tunnel command output now includes information from the hardware counters for each tunnel. For example:

```
IP GRE Tunnel 1 HW Counters:
  InOctets      0      OutOctets      0
  InPkts        0      OutPkts        0
```

Syntax: `show statistics tunnel [tunnel-ID]`

The *tunnel-ID* variable specifies the tunnel ID number.

TABLE 11 show ip tunnel traffic output descriptions

Field	Description
Tunnel Status	Indicates whether the tunnel is up or down. Possible values are: <ul style="list-style-type: none"> Up/Up - The tunnel and line protocol are up. Up/Down - The tunnel is up and the line protocol is down. Down/Up - The tunnel is down and the line protocol is up. Down/Down - The tunnel and line protocol are down.
Packet Received	The number of packets received on the tunnel since it was last cleared by the administrator.
Packet Sent	The number of packets sent on the tunnel since it was last cleared by the administrator.
KA rcv	The number of keepalive packets received on the tunnel since it was last cleared by the administrator.
KA sent	The number of keepalive packets sent on the tunnel since it was last cleared by the administrator.

Displaying multicast protocols and GRE tunneling information

The following **show** commands display information about multicast protocols and GRE tunnels:

- **show ip pim interface**
- **show ip pim nbr**
- **show ip pim mcache**
- **show ip pim flow**
- **show statistics**
- **show ip mtu**

NOTE

All other **show** commands that are supported currently for Ethernet, VE, and IP loopback interfaces, are also supported for tunnel interfaces. To display information for a tunnel interface, specify the tunnel in the format **tn num** . For example, **show interface tn 1**. In some cases, the Ethernet port that the tunnel is using will be displayed in the format **tnnum:eport** .

The following shows an example output of the **show ip pim interface** command.

```
device# show ip pim interface
Interface e1
PIM Dense: V2
TTL Threshold: 1, Enabled, DR: itself
Local Address: 10.10.10.10
Interface tn1
PIM Dense: V2
TTL Threshold: 1, Enabled, DR: 10.1.1.20 on tn1:e2
Local Address: 10.1.1.10
```

```
Neighbor:
 10.1.1.20
```

Syntax: show ip pim interface

The following shows an example output of the **show ip pim nbr** command.

```
device# show ip pim nbr
Total number of neighbors: 1 on 1 ports
Port   Phy_p   Neighbor   Holdtime Age   UpTime
tn1    tn1:e2   10.1.1.20   180     60   1740
```

Syntax: show ip pim nbr

The following shows an example output of the **show ip pim mcache** command.

```
device# show ip pim mcache 230.1.1.1
1      (10.10.10.1 230.1.1.1) in e1 (e1), cnt=629
      Source is directly connected

L3 (HW) 1: tn1:e2(VL1)
      fast=1 slow=0 pru=1 graft
      age=120s up-time=8m HW=1 L2-vidx=8191 has mll
```

Syntax: show ip pim mcache ip-address

The following shows an example output of the **show ip pim flow** command.

```
device# show ip pim flow 230.1.1.1
Multicast flow (10.10.10.1 230.1.1.1):
Vidx for source vlan forwarding: 8191 (Blackhole, no L2 clients)
Hardware MC Entry hit on devices: 0 1 2 3
MC Entry[0x0c008040]: 00014001 000022ee 0ffc0001 00000000
--- MLL contents read from Device 0 ---
MLL Data[0x018c0010]: 0021ff8d 00000083 00000000 00000000
First : Last:1, outlif:60043ff1 00000000, TNL:1(e2)
1 flow printed
```

Syntax: show ip pim flow

The following shows an example output of the **show statistics** command. The following statistics demonstrate an example where the encapsulated multicast traffic ingresses a tunnel endpoint on port e 2, egresses and re-ingresses as native multicast traffic on the loopback port e 4, and is then forwarded to the outbound interface e 1.

```
device# show statistics
Port      In Packets      Out Packets      In Errors      Out Errors
1          0              1670             0              0
2        1668              7               0              0
3          0              0               0              0
4        1668             1668             0              0
```

Syntax: show statistics

The **show ip mtu** command can be used to see if there is space available for the `ip_default_mtu_24` value in the system, or if the MTU value is already configured in the IP table. The following shows an example output of the **show ip mtu** command.

```
device(config-tnif-10)#show ip mtu
idx  size  usage  ref-count
0   10218    1    default
1     800    0         1
2     900    0         1
3     750    0         1
4   10194    1         1
5   10198    0         1
```

Syntax: show ip mtu

Clearing GRE statistics

Use the **clear ip tunnel** command to clear statistics related to GRE tunnels.

To clear GRE tunnel statistics, enter a command such as the following.

```
device(config)# clear ip tunnel stat 3
```

To reset a dynamically-configured MTU on a tunnel Interface back to the configured value, enter a command such as the following.

```
device(config)#clear ip tunnel pmtud 3
```

Syntax: **clear ip tunnel** { **pmtud** *tunnel-ID* | **stat** *tunnel-ID* }

Use the **pmtud** option to reset a dynamically-configured MTU on a tunnel Interface back to the configured value.

Use the **stat** option to clear tunnel statistics.

The *tunnel-ID* variable is a valid tunnel number or name.

Use the **clear statistics tunnel** command to clear GRE tunnel statistics for a specific tunnel ID number. To clear GRE tunnel statistics for tunnel ID 3, enter a command such as the following.

```
device(config)# clear statistics tunnel 3
```

Syntax: **clear statistics tunnel** [*tunnel-ID*]

The *tunnel-ID* variable specifies the tunnel ID number.

Bandwidth for IP interfaces

The bandwidth for an IP interface can be specified so that higher level protocols, such as OSPFv2 and OSPFv3, can use this setting to influence the routing cost for routes learned on these interfaces.

When the interface bandwidth is configured, the number of network and router link state advertisement generation is reduced during an operation down or a shutdown of one or more of the associated interfaces of the VE interface. For OSPF, when the dynamic cost feature is enabled, the bandwidth for a VE interface is the sum of bandwidth for either all associated ports or all active associated ports. However, when the interface bandwidth is configured on the VE interface itself, the bandwidth of the associated ports are not used in the OSPF cost calculation. This means that even when one of the associated ports of the VE interface goes down, there is no OSPF cost recalculation.

The bandwidth for IP interfaces feature can be configured for a physical interface, Link aggregation (LAG) groups, a VE interface, and a tunnel interface.

The bandwidth for IP interfaces feature can be used to:

- Query the bandwidth for an interface.
- Help OSPF avoid generating numerous LSAs while updating the cost value for a VE interface due to changes in associated physical interfaces.
- Influence the cost on OSPF interfaces for specific tunnels, VE interfaces, and physical interfaces.

The bandwidth for IP interfaces feature enables OSPF to calculate its interface metric cost more precisely, based on the specified interface bandwidth. If the interface bandwidth feature is disabled, OSPF calculates the cost as the reference-bandwidth divided by the fixed port bandwidth, as outlined in the [Changing the reference bandwidth for the cost on OSPFv2 interfaces](#) on page 255 section.

When the interface bandwidth feature is enabled, OSPF calculates the cost as the reference-bandwidth divided by the interface bandwidth. For a physical interface, the interface bandwidth is assigned by default to the port speed.

The interface bandwidth feature also enables OSPF to use the configured interface bandwidth for a VE interface to calculate its routing metric, without considering the bandwidth of the associated physical ports. When this feature is enabled, the bandwidth for a VE interface is the interface bandwidth value if it is configured under the VE. Alternatively, it is the sum of the interface bandwidth for all associated ports or all active ports when OSPF dynamic cost is enabled.

The bandwidth of a trunk port for OSPF is, by default, the sum of either all the associated ports or all active associated ports when OSPF dynamic cost is enabled. The interface bandwidth of the primary port is used if the interface bandwidth is configured; otherwise it reverts to the default behavior.

NOTE

If the interface bandwidth configuration of the primary port is different to any of the secondary ports, then the LAG is not deployed. When the LAG is undeployed, the interface bandwidth value for all secondary ports is reset to the port speed.

The configured value is exposed in SNMP via ifSpeed (in ifTable) and ifHighSpeed (in ifXTable) objects.

NOTE

GRE or IPv6 tunnel bandwidth may limit routing protocol traffic propagating through the tunnel. For example, if the tunnel defaults to 8kbps, OSPF uses 50% of the tunnel bandwidth for Hello and update traffic. Therefore, it is good practice to increase the tunnel bandwidth when a routing protocol runs over it to eliminate flapping, and give the routing protocol more capacity to send its update and Hello messages.

From FastIron Release 08.0.30, this feature is supported on all platforms.

Limitations and pre-requisites

- The bandwidth for IP interfaces feature does not support setting and adjusting GRE or IPv6 receiving and transmission bandwidth.
- SNMP does not support any IP interface bandwidth related configurations.

OSPF cost calculation with interface bandwidth

OSPF uses a formula to calculate a path cost when interface bandwidth is available.

If the interface bandwidth feature is disabled, OSPF calculates the cost as the reference-bandwidth divided by the fixed port bandwidth, as outlined in the [Changing the reference bandwidth for the cost on OSPFv2 interfaces](#) on page 255 section. When the interface bandwidth feature is enabled, OSPF calculates the cost as the reference-bandwidth divided by the interface bandwidth.

OSPF uses the following formula to calculate the path cost when interface bandwidth is available:

- OSPF path cost = ((auto-cost × reference-bandwidth + interface bandwidth) - 1) / interface bandwidth.

In the above formula, the cost is calculated in megabits per second (Mbps). The auto-cost is configured using the **auto-cost reference-bandwidth** command in OSPF router configuration mode or OSPFv3 router configuration mode. For more information on changing the OSPF auto-cost reference-bandwidth, refer to the [Changing the reference bandwidth for the cost on OSPFv3 interfaces](#) on page 281 section.

Setting the bandwidth value for an Ethernet interface

The current bandwidth value for an Ethernet interface can be set and communicated to higher-level protocols such as OSPF.

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

```
device# configure terminal
```

Bandwidth for IP interfaces

2. Enter the **interface ethernet** command to configure an Ethernet interface and enter interface configuration mode.

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

3. Enter the **bandwidth** command and specify a value to set the bandwidth value on the interface.

```
device(config-if-e1000-1/1/1)# bandwidth 2000
```

This example sets the bandwidth to 2000 kbps on a specific Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# bandwidth 2000
```

The bandwidth specified in this example results in the following OSPF cost, assuming the auto-cost is 100:

- OSPF cost is equal to $((100 * 1000) + (2000 - 1) / 2000) = 50$

Setting the bandwidth value for a VE interface

The current bandwidth value for a VE interface can be set and communicated to higher-level protocols such as OSPF.

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

```
device# configure terminal
```

2. Enter the **vlan** command and specify a value to configure a VLAN.

```
device(config)# vlan 10
```

3. Enter the **tagged ethernet** command and specify an interface to add a port that is connected to the device and host in the same port-based VLAN.

```
device(config-vlan-10)# tagged ethernet 1/1/1
```

4. Enter the **router-interface ve** command and specify a value to create a virtual interface as the routing interface for the VLAN.

```
device(config-vlan-10)# router-interface ve 10
```

Creates VE 10 as the routing interface for the VLAN.

5. Enter the **interface ve** command and specify a value.

```
device(config-vlan-10)# interface ve 10
```

Creates a VE interface with the VLAN ID of 10.

6. Enter the **bandwidth** command and specify a value to set the bandwidth value on the interface.

```
device(config-vif-10)# bandwidth 2000
```

This example sets the bandwidth to 2000 kbps on a specific VE interface .

```
device# configure terminal
device(config)# vlan 10
device(config-vlan-10)# tagged ethernet 1/1/1
device(config-vlan-10)# router-interface ve 10
device(config-vlan-10)# interface ve 10
device(config-vif-10)# bandwidth 2000
```

The bandwidth specified in this example results in the following OSPF cost, assuming the auto-cost is 100:

- OSPF cost is equal to $((100 * 1000) + (2000 - 1) / 2000) = 50$

Setting the bandwidth value for a tunnel interface

The current bandwidth value for a tunnel interface can be set and communicated to higher-level protocols such as OSPF.

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

```
device# configure terminal
```

2. Enter the **interface tunnel** command and specify a value to configure a tunnel interface.

```
device(config)# interface tunnel 2
```

3. Enter the **tunnel mode gre ip** command to enable GRE IP encapsulation on the tunnel interface.

```
device(config-tnif-2)# tunnel mode gre ip
```

4. Enter the **tunnel source** command and specify an IP address to configure the source address for the tunnel interface.

```
device(config-tnif-2)# tunnel source 10.0.0.1
```

5. Enter the **tunnel destination** command and specify an IP address to configure the destination address for the tunnel interface.

```
device(config-tnif-2)# tunnel destination 10.10.0.1
```

6. Enter the **ip address** command and specify an IP address and a network mask to assign an IP address to the tunnel interface.

```
device(config-tnif-2)# ip address 10.0.0.1/24
```

7. Enter the **bandwidth** command and specify a value to set the bandwidth value on the interface.

```
device(config-tnif-2)# bandwidth 2000
```

This example sets the bandwidth to 2000 kbps on a specific tunnel interface .

```
device# configure terminal
device(config)# interface tunnel 2
device(config-tnif-2)# tunnel mode gre ip
device(config-tnif-2)# tunnel source 10.0.0.1
device(config-tnif-2)# tunnel destination 10.10.0.1
device(config-tnif-2)# ip address 10.0.0.1/24
device(config-tnif-2)# bandwidth 2000
```

The bandwidth specified in this example results in the following OSPF interface costs, assuming the auto-cost is 100:

- OSPF Interface Cost for the Trunk Group is equal to $((100 * 1000) + (2000 - 1) \div 2000) = 50$
- OSPF Interface Cost for the GRE/IPv6 tunnel is equal to $((100 * 1000) + (2000 - 1) \div 2000) = 50$

User-configurable MAC address per IP interface

Manual configuration of one IP MAC address for each Layer 3 physical or virtual ethernet (VE) interface on a device is permitted. The configured MAC address is used as the source MAC address by routing protocols or hardware communication related to the IPv4 or IPv6 addresses on the interface, for example in ARP or neighbor discovery (ND) packets to the interface. The IPv4 and IPv6 addresses use the same IP MAC address for any software and hardware communication.

If an IP MAC address is not configured, the IP interface uses the MAC address from the router or stack.

The user-configurable IP MAC address feature supports the following unicast protocols:

- IPv4 support—ARP, BGP, OSPF, RIP

- IPv6 support—BGP4+, Neighbor Discovery (ND), OSPFv3, RD, RIPng

In addition to the unicast protocol support, the configured MAC address is used by IPv4 and IPv6 unicast software-generated packets (for example, ping) and IPv4 and IPv6 hardware-forwarded packets. For IPv4 addresses that are configured on the IP interface, gratuitous ARP is generated when the IP MAC address is configured. For IPv6 addresses, DAD is started and link-local addresses are regenerated when the IP MAC address is configured.

If Virtual Router Redundancy Protocol (VRRP) IPv4 or IPv6 sessions are configured on an interface where an IP MAC is configured, the VRRP sessions continue to use the virtual MAC address assigned to the virtual router ID (VRID) for any ARP or ND queries.

Some restrictions apply to the user-configurable MAC address per IP interface feature:

- The manually configured IP MAC address is not supported for multicast communications.
- The IP MAC address must be unique on the device including any interfaces. If the device is configured as part of a stack, the IP MAC address must not be the same as the MAC address of other stack units. If a stack MAC address is configured it must not be the same as the IP MAC on any interface.
- The IP MAC address configured manually for a VE interface must be unique within the same VLAN.
- There is a maximum number of IP interfaces (248) on which an IP MAC address can be configured and the number of VRRP virtual interfaces that can be supported simultaneously is affected by any increase over the default number of 120 interfaces. If the **system-max max-ip-mac** command is set above 120, a reduction in the number of IPv4 VRRP entries supported is calculated as <configured-value> - 120. For example, if the **system-max max-ip-mac** value is set to 130, the number of IPv4 VRRP entries is reduced by 10 entries (130-120).

Manually configuring an IP MAC address

One IP MAC address can be manually configured for each Layer 3 physical or virtual ethernet (VE) interface on a device. The configured MAC address will be used for all the software and hardware communications related to unicast IPv4 or IPv6 addresses on the IP interface.

1. From privileged EXEC mode, enter configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Configure an ethernet interface link.

```
device(config)# interface ethernet 1/1/6
```

3. Configure the IP address of the interface.

```
device(config-if-e1000-1/1/6)# ip address 10.53.5.1/24
```

4. Configure a MAC address on the interface.

```
device(config-if-e1000-1/1/6)# ip-mac aaaa.bbbb.cccc
```

5. Exit to Privileged EXEC mode.

```
device(config-if-e1000-1/1/6)# end
```


6. Use the **show ip interface** command to verify the user-configured MAC address.

```
device# show ip interface ethernet 1/1/6

Interface Ethernet 1/1/6
port enabled
port state: DOWN
ip address: 10.53.5.1          subnet mask: 255.255.255.0
Port belongs to VRF: default-vrf
encapsulation: ETHERNET, mtu: 1500, metric: 1
directed-broadcast-forwarding: disabled
ICMP redirect: disabled
proxy-arp: disabled
ip arp-age: 10 minutes
No Helper Addresses are configured.
No inbound ip access-list is set
No outgoing ip access-list is set
ip-mac: aaaa.bbbb.cccc
```

The user-configured MAC address is shown after the "ip-mac:" text.

Modifying and displaying Layer 3 system parameter limits

This section shows how to view and configure some of the Layer 3 system parameter limits.

Layer 3 configuration notes

- Changing the system parameters reconfigures the device memory. Whenever you reconfigure the memory on a Brocade device, you must save the change to the startup-config file, and then reload the software to place the change into effect.
- The Layer 3 system parameter limits for FastIron IPv6 models are automatically adjusted by the system and cannot be manually modified.

Displaying Layer 3 system parameter limits

To display the Layer 3 system parameter defaults, maximum values, and current values, enter the **show default value** command at any level of the CLI.

The following example shows output on an ICX 7450 with third generation modules.

```
device# show default value

sys log buffers:50          mac age time:300 sec          telnet sessions:5
ip arp age:10 min          bootp relay max hops:4        ip ttl:64 hops
ip addr per intf:24

when multicast enabled :   igmp query:125 sec          hardware drop: enabled
igmp group memb.:260 sec

when ospf enabled :       ospf hello:10 sec           ospf retrans:5 sec
ospf dead:40 sec
ospf transit delay:1 sec

when bgp enabled :        bgp keep alive:60 sec        bgp hold:180 sec
bgp local pref.:100       bgp local as:1              bgp cluster id:0
bgp metric:10             bgp int. distance:200       bgp local distance:200
bgp ext. distance:20
```

Enabling or disabling routing protocols

System Parameters	Default	Maximum	Current	Configured
ip-arp	4000	64000	4000	4000
ip-static-arp	512	6000	512	512
ip-cache	10000	32768	10000	10000
ip-filter-port	3071	3071	3071	3071
ip-filter-sys	3072	8192	3072	3072
l3-vlan	32	1024	32	32
ip-qos-session	1024	16000	1024	1024
mac	32768	32768	32768	32768
ip-route	12000	15168	12000	12000
ip-static-route	64	2048	64	64
some lines omitted for brevity....				
dot1x-mka-policy-gro	8	8	8	8
openflow-flow-entrie	3072	12288	3072	3072
openflow-pvlan-entri	40	40	40	40
openflow-unprotected	40	40	40	40
openflow-nexthop-ent	1024	3072	1024	1024
max-ip-mac	128	256	128	128
max-dhcp-snoop-entri	1024	3072	1024	1024
max-static-inspect-a	512	1024	512	512

Enabling or disabling routing protocols

This section describes how to enable or disable routing protocols. For complete configuration information about the routing protocols, refer to the respective chapters in this guide.

The Layer 3 code supports the following protocols:

- BGP4
- IGMP
- IP
- IP multicast (PIM-SM, PIM-DM)
- OSPF
- PIM
- RIPV1 and V2
- VRRP
- VRRP-E
- VSRP
- IPv6 Routing
- IPv6 Multicast

IP routing is enabled by default on devices running Layer 3 code. All other protocols are disabled, so you must enable them to configure and use them.

To enable a protocol on a device running Layer 3 code, enter **router** at the global CONFIG level, followed by the protocol to be enabled. The following example shows how to enable OSPF.

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

Syntax: router bgp | igmp | ip | ospf | pim | rip | vrrp | vrrp-e | vsrp

Enabling or disabling Layer 2 switching

By default, Brocade Layer 3 switches support Layer 2 switching. These devices modify the routing protocols that are not supported on the devices. If you want to disable Layer 2 switching, you can do so globally or on individual ports, depending on the version of software your device is running.

NOTE

Consult your reseller or Brocade to understand the risks involved before disabling all Layer 2 switching operations.

Configuration notes and feature limitations for Layer 2 switching

- Enabling or disabling Layer 2 switching is supported in Layer 3 software images only.
- Brocade ICX devices support disabling Layer 3 switching at the interface configuration mode as well as the global configuration mode.
- Enabling or disabling Layer 2 switching is not supported on virtual interfaces.

Command syntax for Layer 2 switching

To globally disable Layer 2 switching on a Layer 3 switch, enter commands such as the following.

```
device(config)#route-only
device(config)#exit
device#write memory
device#reload
```

To re-enable Layer 2 switching on a Layer 3 switch, enter the following commands.

```
device(config)#no route-only
device(config)#exit
device#write memory
device#reload
```

Syntax: [no] route-only

To disable Layer 2 switching only on a specific interface, go to the interface configuration level for that interface, and then disable the feature. The following commands show how to disable Layer 2 switching on port 2.

```
device(config)#interface ethernet 2
device(config-if-e1000-2)#route-only
```

Configuring a Layer 3 Link Aggregation Group (LAG)

Configuring a Layer 3 Link Aggregation Group (LAG)

FastIron devices with Layer 3 images support Layer 3 LAGs, which are used for routing and not switching. For details on how to create a LAG, refer to *Link Aggregation* in the *Brocade FastIron Platform and Layer 2 Switching Configuration Guide*. Perform the following steps to enable routing on a LAG:

1. In the global configuration mode, run the **interface ethernet** command to enter the interface configuration mode of the primary port of the LAG.

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

Disabling IP checksum check

2. Run the **route-only** command to disable switching and enable routing on the LAG.

```
Brocade(config-if-e1000-4/1/4)# route-only
```

3. Run the **ip address** command to assign an IP address for the LAG.

```
Brocade(config-if-e1000-4/1/4)# ip address 25.0.0.2/24
```

The following example shows the creation and deployment of a dynamic LAG that is used for routing on a FastIron device with Layer 3 image.

```
Brocade(config)# lag "brocade-LAG" dynamic id 55
Brocade(config-lag- brocade-LAG)# ports ethernet 1/1/1 ethernet 2/1/3 ethernet 3/1/4 ethernet 4/1/4
Brocade(config-lag- brocade-LAG)# primary-port 4/1/4
Brocade(config-lag- brocade-LAG)# deploy
Brocade(config-lag- brocade-LAG)# exit
Brocade(config)# interface ethernet 4/1/4
Brocade(config-if-e1000-4/1/4)# route-only
Brocade(config-if-e1000-4/1/4)# ip address 25.0.0.2/24
```

Disabling IP checksum check

The **disable-hw-ip-checksum-check** command traps a packet with bad checksum to the CPU. Previously, if the packet processor detected a packet with, for example, the checksum 0xFFFF, it would treat it as a bad checksum even if it was correct and it would drop the packet. Now, the command **disable-hw-ip-checksum-check** traps the packet at the CPU and if the checksum is correct, it forwards the packet.

To set disable hardware ip checksum check for all ports, enter the following command.

```
device# disable-hw-ip-checksum-check
disable-ip-header-check set for all ports
```

To clear disable hardware ip checksum check on all ports, enter the following command.

```
device# no disable-hw-ip-checksum-check ethernet 13
disable-hw-ip-checksum-check cleared for ports the 13 to 24
```

To set disable hardware ip checksum check on for example, port range 0-12, enter the following command.

```
device# disable-hw-ip-checksum-check ethernet 2
disable-ip-header-check set for ports ethe 1 to 12
```

To set disable hardware ip checksum check on, for example, port range 13-24, enter the following command.

```
device# disable-hw-ip-checksum-check ethernet 22
disable-ip-header-check set for ports ethe 13 to 24
```

To clear disable hardware ip checksum check on, for example, port range 13-24, enter the following command.

```
device# no disable-hw-ip-checksum-check ethernet 13
disable-hw-ip-checksum-check cleared for ports the 13 to 24
```

NOTE

The port range could be any consecutive range, it may not necessarily be a decimal number.

Syntax: **[no] disable-hw-ip-checksum-check ethernet *portnum***

NOTE

This command only functions on the IPv4 platform.

Displaying IP configuration information and statistics

The following sections describe IP display options for Layer 3 switches and Layer 2 switches.

By default, the CLI displays network masks in classical IP address format (example: 255.255.255.0). You can change the displays to prefix format (example: /18) on a Layer 3 switch or Layer 2 switch using the following CLI method.

NOTE

This option does not affect how information is displayed in the Web Management Interface.

To enable CIDR format for displaying network masks, entering the following command at the global CONFIG level of the CLI.

```
device(config)# ip show-subnet-length
```

Displaying IP information - Layer 3 switches

You can display the following IP configuration information statistics on Layer 3 switches:

- Global IP parameter settings and IP access policies
- CPU utilization statistics
- IP interfaces
- ARP entries
- Static ARP entries
- IP forwarding cache
- IP route table
- IP traffic statistics

Displaying global IP configuration information

To display IP configuration information, enter the following command at any CLI level.

```
device# show ip
Global Settings
  ttl: 64, arp-age: 10, bootp-relay-max-hops: 4
  router-id : 10.95.11.128
enabled : UDP-Broadcast-Forwarding Source-Route Load-Sharing RARP OSPF VRRP-Extended VSRP
disabled: Route-Only Directed-Broadcast-Forwarding BGP4 IRDP Proxy-ARP RIP VRRP ICMP-Redirect
Static Routes
  Index  IP Address      Subnet Mask      Next Hop Router  Metric Distance
  1      0.0.0.0          0.0.0.0          10.157.23.2      1      1
Policies
  Index  Action  Source      Destination      Protocol  Port  Operator
  1      deny   any         10.157.22.34     10.157.22.26    tcp
  64     permit any         any
```

Syntax: show ip

NOTE

This command has additional options, which are explained in other sections in this guide, including the sections following this one.

This display shows the following information.

TABLE 12 CLI display of global IP configuration information - Layer 3 switch

Field	Description
Global settings	
ttl	The Time-To-Live (TTL) for IP packets. The TTL specifies the maximum number of router hops a packet can travel before reaching the Brocade router. If the packet TTL value is higher than the value specified in this field, the Brocade router drops the packet.
arp-age	The ARP aging period. This parameter specifies how many minutes an inactive ARP entry remains in the ARP cache before the router ages out the entry.
bootp-relay-max-hops	The maximum number of hops away a BootP server can be located from the Brocade router and still be used by the router clients for network booting.
router-id	The 32-bit number that uniquely identifies the Brocade router. By default, the router ID is the numerically lowest IP interface configured on the router.
enabled	The IP-related protocols that are enabled on the router.
disabled	The IP-related protocols that are disabled on the router.
Static routes	
Index	The row number of this entry in the IP route table.
IP Address	The IP address of the route destination.
Subnet Mask	The network mask for the IP address.
Next Hop Router	The IP address of the router interface to which the Brocade router sends packets for the route.
Metric	The cost of the route. Usually, the metric represents the number of hops to the destination.
Distance	The administrative distance of the route. The default administrative distance for static IP routes in Brocade routers is 1.
Policies	
Index	The policy number. This is the number you assigned the policy when you configured it.
Action	The action the router takes if a packet matches the comparison values in the policy. The action can be one of the following: <ul style="list-style-type: none"> deny - The router drops packets that match this policy. permit - The router forwards packets that match this policy.
Source	The source IP address the policy matches.
Destination	The destination IP address the policy matches.
Protocol	The IP protocol the policy matches. The protocol can be one of the following: <ul style="list-style-type: none"> ICMP IGMP IGRP OSPF TCP UDP
Port	The Layer 4 TCP or UDP port the policy checks for in packets. The port can be displayed by its number or, for port types the router recognizes, by

TABLE 12 CLI display of global IP configuration information - Layer 3 switch (continued)

Field	Description
	the well-known name. For example, TCP port 80 can be displayed as HTTP. NOTE This field applies only if the IP protocol is TCP or UDP.
Operator	The comparison operator for TCP or UDP port names or numbers. NOTE This field applies only if the IP protocol is TCP or UDP.

Displaying IP interface information

To display IP interface information, enter the following command at any CLI level.

```
device# show ip interface
Interface      IP-Address      OK?  Method      Status      Protocol
Ethernet 1/1/1  10.95.6.173     YES  NVRAM       up          up
Ethernet 1/1/2  10.3.3.3        YES  manual      up          up
Loopback 1     10.2.3.4        YES  NVRAM       down       down
```

Syntax: `show ip interface [ethernet unit / slot / port | loopback num | tunnel num | venum]`

This display shows the following information.

TABLE 13 CLI display of interface IP configuration information

Field	Description
Interface	The type and the slot and port number of the interface.
IP-Address	The IP address of the interface. NOTE If an "s" is listed following the address, this is a secondary address. When the address was configured, the interface already had an IP address in the same subnet, so the software required the "secondary" option before the software could add the interface.
OK?	Whether the IP address has been configured on the interface.
Method	Whether the IP address has been saved in NVRAM. If you have set the IP address for the interface in the CLI or Web Management Interface, but have not saved the configuration, the entry for the interface in the Method field is "manual".
Status	The link status of the interface. If you have disabled the interface with the disable command, the entry in the Status field will be "administratively down". Otherwise, the entry in the Status field will be either "up" or "down".
Protocol	Whether the interface can provide two-way communication. If the IP address is configured, and the link status of the interface is up, the entry in the protocol field will be "up". Otherwise the entry in the protocol field will be "down".

To display detailed IP information for a specific interface, enter a command such as the following.

```
device# show ip interface ve 1
Interface Ve 1
```

Displaying IP configuration information and statistics

```
members: ethe 1/1/4 to 1/1/24 ethe 1/1/27 to 1/1/48 ethe 1/2/1 to 1/2/2 ethe 2/1/1 to 2/1/2
ethe 2/1/4 to 2/1/12 ethe 2/1/15 to 2/1/24 ethe 2/2/1 to 2/2/2 ethe 3/1/1 to 3/1/2 ethe 3/1/4 to 3/1/12
ethe 3/1/14 to 3/1/24 ethe 3/2/3 to 3/2/4 ethe 4/1/1 to 4/1/12 ethe 4/1/15 to 4/1/24 ethe 4/2/3 to 4/2/4
ethe 5/1/1 to 5/1/12 ethe 5/1/14 to 5/1/24 ethe 5/2/3
active: ethe 4/2/4
port enabled
port state: UP
ip address: 66.66.66.66          subnet mask: 255.255.255.0
Port belongs to VRF: default-vrf
encapsulation: ETHERNET, mtu: 9216, metric: 1
directed-broadcast-forwarding: disabled
ICMP redirect: enabled
proxy-arp: disabled
ip arp-age: 10 minutes
No Helper Addresses are configured.
No inbound ip access-list is set
No outgoing ip access-list is set
```

Displaying ARP entries

You can display the ARP cache and the static ARP table. The ARP cache contains entries for devices attached to the Layer 3 switch. The static ARP table contains the user-configured ARP entries. An entry in the static ARP table enters the ARP cache when the entry interface comes up.

The tables require separate display commands or Web management options.

Displaying the ARP cache

To display the contents of the ARP cache, enter the following command at any CLI level.

```
Brocade# show arp
Total number of ARP entries: 70
Entries in default routing instance:
No.   IP Address      MAC Address      Type      Age  Port          Status
1     10.63.61.2        000c.000c.000c   Dynamic   0    1/1/16-1/1/17 Valid
2     10.63.53.2        000c.000c.000c   Dynamic   0    1/1/16-1/1/17 Valid
3     10.63.45.2        000c.000c.000c   Dynamic   0    1/1/16-1/1/17 Valid
4     10.63.37.2        000c.000c.000c   Dynamic   0    1/1/16-1/1/17 Valid
5     10.63.29.2        000c.000c.000c   Dynamic   0    1/1/16-1/1/17 Valid
6     10.63.21.2        000c.000c.000c   Dynamic   0    1/1/16-1/1/17 Valid
7     10.63.13.2        000c.000c.000c   Dynamic   0    1/1/16-1/1/17 Valid
8     10.63.0.1         000c.000c.000c   Dynamic   0    1/1/16-1/1/17 Valid
9     10.63.5.2         000c.000c.000c   Dynamic   0    1/1/16-1/1/17 Valid
10    10.63.62.2        000c.000c.000c   Dynamic   0    1/1/16-1/1/17 Valid
11    10.63.54.2        000c.000c.000c   Dynamic   0    1/1/16-1/1/17 Valid
--More--
```

To display the contents of the ARP cache when a VRF is configured, enter the following command at any CLI level.

```
Brocade# show arp vrf one
Total number of ARP entries: 1
Entries in VRF one:
No.   IP Address      MAC Address      Type      Age  Port          Status
1     10.65.0.2        000c.000c.000c   Dynamic   1    1/1/16-1/1/17 Valid
```

Syntax: `show arp [ip-addr [ip-mask] | num-entries-to-skip | ethernet unit / slot / port | inspect | mac-address xxxx.xxxx.xxxx [MAC-mask] | management man-port | resource | vrf vrf-name]`

The `ip-addr` and `ip-mask` parameters let you restrict the display to entries for a specific IP address and network mask. Specify the IP address masks in standard decimal mask format (for example, 255.255.0.0).

The `mac-address xxxx.xxxx.xxxx` parameter lets you restrict the display to entries for a specific MAC address.

The `MAC-mask` parameter lets you specify a mask for the `mac-address xxxx.xxxx.xxxx` parameter, to display entries for multiple MAC addresses. Specify the MAC address mask as "f"s and "0"s, where "f"s are significant bits.

NOTE

The *ip-mask* parameter and *mask* parameter perform different operations. The *ip-mask* parameter specifies the network mask for a specific IP address, whereas the *mask* parameter provides a filter for displaying multiple MAC addresses that have specific values in common.

The **vrf** *vrf-name* parameter lets you restrict the display to entries for a specific VRF.

The *num-entries-to-skip* parameter lets you display the table beginning with a specific entry number.

NOTE

The entry numbers in the ARP cache are not related to the entry numbers for static ARP table entries.

This display shows the following information. The number in the left column of the CLI display is the row number of the entry in the ARP cache. This number is not related to the number you assign to static MAC entries in the static ARP table.

TABLE 14 CLI display of ARP cache

Field	Description
Total number of ARP Entries	The number of entries in the ARP cache.
Entries in default routing instance	The total number of ARP entries supported on the device.
Entries in VRF vrf-name	The total number of ARP entries for the specified VRF.
IP Address	The IP address of the device.
MAC Address	The MAC address of the device.
Type	<p>The ARP entry type, which can be one of the following:</p> <ul style="list-style-type: none"> Dynamic - The Layer 3 switch learned the entry from an incoming packet. Static - The Layer 3 switch loaded the entry from the static ARP table when the device for the entry was connected to the Layer 3 switch. DHCP - The Layer 3 Switch learned the entry from the DHCP binding address table. <p>NOTE If the type is DHCP, the port number will not be available until the entry gets resolved through ARP.</p>
Age	<p>The number of minutes before which the ARP entry was refreshed. If this value reaches the ARP aging period, the entry is removed from the table.</p> <p>NOTE Static entries do not age out.</p>
Port	<p>The port on which the entry was learned.</p> <p>NOTE If the ARP entry type is DHCP, the port number will not be available until the entry gets resolved through ARP.</p>
Status	<p>The status of the entry, which can be one of the following:</p> <ul style="list-style-type: none"> Valid - This a valid ARP entry. Pend - The ARP entry is not yet resolved.

Displaying IP configuration information and statistics

Displaying the static ARP table

To display the static ARP table instead of the ARP cache, enter the following command at any CLI level.

```
device# show ip static-arp
Static ARP table size: 512, configurable from 512 to 1024
Index IP Address      MAC Address      Port
 1    10.95.6.111      0000.003b.d210   1/1/1
 3    10.95.6.123      0000.003b.d211   1/1/1
```

This example shows two static entries. Note that because you specify an entry index number when you create the entry, it is possible for the range of index numbers to have gaps, as shown in this example.

NOTE

The entry number you assign to a static ARP entry is not related to the entry numbers in the ARP cache.

Syntax: `show ip static-arp [ip-addr [ip-mask] | num-entries-to-skip | ethernet unit / slot / port | mac-address xxxx.xxxx.xxxx [MAC-mask]]`

The *ip-addr* and *ip-mask* parameters let you restrict the display to entries for a specific IP address and network mask. Specify the IP address masks in standard decimal mask format (for example, 255.255.0.0).

The **mac-address**xxxx.xxxx.xxxx parameter lets you restrict the display to entries for a specific MAC address.

The *mask* parameter lets you specify a mask for the **mac-address**xxxx.xxxx.xxxx parameter, to display entries for multiple MAC addresses. Specify the MAC address mask as "f"s and "O"s, where "f"s are significant bits.

NOTE

The *ip-mask* parameter and *mask* parameter perform different operations. The *ip-mask* parameter specifies the network mask for a specific IP address, whereas the *mask* parameter provides a filter for displaying multiple MAC addresses that have specific values in common.

The *num-entries-to-skip* parameter lets you display the table beginning with a specific entry number.

TABLE 15 CLI display of static ARP table

Field	Description
Static ARP table size	The maximum number of static entries that can be configured on the device using the current memory allocation. The range of valid memory allocations for static ARP entries is listed after the current allocation.
Index	The number of this entry in the table. You specify the entry number when you create the entry.
IP Address	The IP address of the device.
MAC Address	The MAC address of the device.
Port	The port attached to the device the entry is for.

Displaying the forwarding cache

To display the IP forwarding cache, enter the following command at any CLI level.

```
device# show ip cache
Total number of cache entries: 3
D:Dynamic P:Permanent F:Forward U:Us C:Complex Filter
W:Wait ARP I:ICMP Deny K:Drop R:Fragment S:Snap Encap
IP Address      Next Hop      MAC          Type  Port  Vlan  Pri
1    192.168.1.11    DIRECT        0000.0000.0000  PU   n/a    0
2    192.168.1.255  DIRECT        0000.0000.0000  PU   n/a    0
3    255.255.255.255 DIRECT        0000.0000.0000  PU   n/a    0
```

Syntax: `show ip cache [ip-addr | num]`

The *ip-addr* parameter displays the cache entry for the specified IP address.

The *num* parameter displays the cache beginning with the row following the number you enter. For example, to begin displaying the cache at row 10, enter the following command.

```
device# show ip cache 9
```

The **show ip cache** command displays the following information.

TABLE 16 CLI display of IP forwarding cache - Layer 3 switch

Field	Description
IP Address	The IP address of the destination.
Next Hop	The IP address of the next-hop router to the destination. This field contains either an IP address or the value DIRECT. DIRECT means the destination is either directly attached or the destination is an address on this Brocade device. For example, the next hop for loopback addresses and broadcast addresses is shown as DIRECT.
MAC	The MAC address of the destination. NOTE If the entry is type U (indicating that the destination is this Brocade device), the address consists of zeroes.
Type	The type of host entry, which can be one or more of the following: <ul style="list-style-type: none"> • D - Dynamic • P - Permanent • F - Forward • U - Us • C - Complex Filter • W - Wait ARP • I - ICMP Deny • K - Drop • R - Fragment • S - Snap Encap
Port	The port through which this device reaches the destination. For destinations that are located on this device, the port number is shown as "n/a".
VLAN	Indicates the VLANs the listed port is in.
Pri	The QoS priority of the port or VLAN.

Displaying the IP route table

To display the IP route table, enter the **show ip route** command at any CLI level.

```
device# show ip route
Total number of IP routes: 514
Start index: 1  B:BGP D:Connected  R:RIP  S:Static  O:OSPF *:Candidate default
Destination      NetMask      Gateway      Port    Cost    Type
10.1.0.0          255.255.0.0  10.1.1.2     1/1/1    2       R
10.2.0.0          255.255.0.0  10.1.1.2     1/1/1    2       R
10.3.0.0          255.255.0.0  10.1.1.2     1/1/1    2       R
10.4.0.0          255.255.0.0  10.1.1.2     1/1/1    2       R
10.5.0.0          255.255.0.0  10.1.1.2     1/1/1    2       R
10.6.0.0          255.255.0.0  10.1.1.2     1/1/1    2       R
10.7.0.0          255.255.0.0  10.1.1.2     1/1/1    2       R
```

Displaying IP configuration information and statistics

10.8.0.0	255.255.0.0	10.1.1.2	1/1/1	2	R
10.9.0.0	255.255.0.0	10.1.1.2	1/1/1	2	R
10.10.0.0	255.255.0.0	10.1.1.2	1/1/1	2	S

Syntax: `show ip route [ip-addr [ip-mask] [longer] [none-bgp]] {num | bgp | direct | ospf | rip | static }`

The *ip-addr* parameter displays the route to the specified IP address.

The *ip-mask* parameter lets you specify a network mask or, if you prefer CIDR format, the number of bits in the network mask. If you use CIDR format, enter a forward slash immediately after the IP address, then enter the number of mask bits (for example: 10.157.22.0/24 for 10.157.22.0 255.255.255.0).

The **longer** parameter applies only when you specify an IP address and mask. This option displays only the routes for the specified IP address and mask. Refer to the following example.

The **none-bgp** parameter displays only the routes that did not come from BGP4.

The *num* option display the route table entry whose row number corresponds to the number you specify. For example, if you want to display the tenth row in the table, enter "10".

The **bgp** option displays the BGP4 routes.

The **direct** option displays only the IP routes that are directly attached to the Layer 3 switch.

The **ospf** option displays the OSPF routes.

The **rip** option displays the RIP routes.

The **static** option displays only the static IP routes.

The **default** routes are displayed first.

Here is an example of how to use the **direct** option. To display only the IP routes that go to devices directly attached to the Layer 3 switch, enter the following command.

```
device# show ip route direct
Start index: 1  B:BGP D:Connected R:RIP S:Static O:OSPF *:Candidate default
Destination      NetMask      Gateway      Port  Cost  Type
10.157.22.0       255.255.255.0  0.0.0.0      1/4/11  1      D
```

Notice that the route displayed in this example has "D" in the Type field, indicating the route is to a directly connected device.

Here is an example of how to use the **static** option. To display only the static IP routes, enter the following command.

```
device# show ip route static
Start index: 1  B:BGP D:Connected R:RIP S:Static O:OSPF *:Candidate default
Destination      NetMask      Gateway      Port  Cost  Type
10.144.33.11      255.255.255.0  10.157.22.12  1/1/1  2      S
```

Notice that the route displayed in this example has "S" in the Type field, indicating the route is static.

Here is an example of how to use the **longer** option. To display only the routes for a specified IP address and mask, enter a command such as the following.

```
device# show ip route 10.159.0.0/16 longer
Starting index: 1  B:BGP D:Directly-Connected R:RIP S:Static O:OSPF
Destination      NetMask      Gateway      Port  Cost  Type
52 10.159.38.0     255.255.255.0  10.95.6.101  1/1/1  1      S
53 10.159.39.0     255.255.255.0  10.95.6.101  1/1/1  1      S
54 10.159.40.0     255.255.255.0  10.95.6.101  1/1/1  1      S
55 10.159.41.0     255.255.255.0  10.95.6.101  1/1/1  1      S
56 10.159.42.0     255.255.255.0  10.95.6.101  1/1/1  1      S
57 10.159.43.0     255.255.255.0  10.95.6.101  1/1/1  1      S
58 10.159.44.0     255.255.255.0  10.95.6.101  1/1/1  1      S
59 10.159.45.0     255.255.255.0  10.95.6.101  1/1/1  1      S
60 10.159.46.0     255.255.255.0  10.95.6.101  1/1/1  1      S
```

This example shows all the routes for networks beginning with 10.159. The mask value and **longer** parameter specify the range of network addresses to be displayed. In this example, all routes within the range 10.159.0.0 - 10.159.255.255 are listed.

The **summary** option displays a summary of the information in the IP route table. The following is an example of the output from this command.

```
device# show ip route summary
IP Routing Table - 35 entries:
 6 connected, 28 static, 0 RIP, 1 OSPF, 0 BGP, 0 ISIS, 0 MPLS
Number of prefixes:
 /0: 1 /16: 27 /22: 1 /24: 5 /32: 1
```

Syntax: show ip route summary

In this example, the IP route table contains 35 entries. Of these entries, 6 are directly connected devices, 28 are static routes, and 1 route was calculated through OSPF. One of the routes has a zero-bit mask (this is the default route), 27 have a 22-bit mask, 5 have a 24-bit mask, and 1 has a 32-bit mask.

The following table lists the information displayed by the **show ip route** command.

TABLE 17 CLI display of IP route table

Field	Description
Destination	The destination network of the route.
NetMask	The network mask of the destination address.
Gateway	The next-hop router. An asterisk (*) next to the next-hop router indicates that it is one of multiple Equal-Cost Multi-Path (ECMP) next hops for a given route. The asterisk will initially appear next to the first next hop for each route with multiple ECMP next hops. If the ARP entry for the <i>next hop</i> * ages out or is cleared, then the next packet to be routed through the Brocade device whose destination matches that route can cause the asterisk to move to the next hop down the list of ECMP next hops for that route. This means that if the <i>next hop</i> * goes down, the asterisk can move to another next hop with equal cost.
Port	The port through which this router sends packets to reach the route's destination.
Cost	The route's cost.
Type	The route type, which can be one of the following: <ul style="list-style-type: none"> • B - The route was learned from BGP. • D - The destination is directly connected to this Layer 3 switch. • R - The route was learned from RIP. • S - The route is a static route. • * - The route and next-hop gateway are resolved through the ip default-network setting. • O - The route is an OSPF route. Unless you use the ospf option to display the route table, "O" is used for all OSPF routes. If you do use the ospf option, the following type codes are used: <ul style="list-style-type: none"> • O - OSPF intra area route (within the same area). • IA - The route is an OSPF inter area route (a route that passes from one area into another). • E1 - The route is an OSPF external type 1 route. • E2 - The route is an OSPF external type 2 route.

Displaying IP configuration information and statistics

Clearing IP routes

If needed, you can clear the entire route table or specific individual routes.

When an interface subnet route with an interface address that directly matches a host route learned from a neighboring device is configured and subsequently removed, the **clear ip route** command should be used so that the learned route is updated in the Routing and Hardware Forwarding table.

To clear all routes from the IP route table, enter the following command.

```
device# clear ip route
```

To clear route 10.157.22.0/24 from the IP routing table, enter the **clear ip route** command.

```
device# clear ip route 10.157.22.0/24
```

Syntax: **clear ip route** [*ip-addr ip-mask*]

or

Syntax: **clear ip route** [*ip-addr/mask-bits*]

Displaying IP traffic statistics

To display IP traffic statistics, enter the **show ip traffic** command at any CLI level.

```
device# show ip traffic
IP Statistics
  139 received, 145 sent, 0 forwarded
  0 filtered, 0 fragmented, 0 reassembled, 0 bad header
  0 no route, 0 unknown proto, 0 no buffer, 0 other errors
ICMP Statistics
Received:
  0 total, 0 errors, 0 unreachable, 0 time exceed
  0 parameter, 0 source quench, 0 redirect, 0 echo,
  0 echo reply, 0 timestamp, 0 timestamp reply, 0 addr mask
  0 addr mask reply, 0 irdp advertisement, 0 irdp solicitation
Sent:
  0 total, 0 errors, 0 unreachable, 0 time exceed
  0 parameter, 0 source quench, 0 redirect, 0 echo,
  0 echo reply, 0 timestamp, 0 timestamp reply, 0 addr mask
  0 addr mask reply, 0 irdp advertisement, 0 irdp solicitation
UDP Statistics
  1 received, 0 sent, 1 no port, 0 input errors
TCP Statistics
  0 active opens, 0 passive opens, 0 failed attempts
  0 active resets, 0 passive resets, 0 input errors
  138 in segments, 141 out segments, 4 retransmission
RIP Statistics
  0 requests sent, 0 requests received
  0 responses sent, 0 responses received
  0 unrecognized, 0 bad version, 0 bad addr family, 0 bad req format
  0 bad metrics, 0 bad resp format, 0 resp not from rip port
  0 resp from loopback, 0 packets rejected
```

The **show ip traffic** command displays the following information.

TABLE 18 CLI display of IP traffic statistics - Layer 3 switch

Field	Description
IP statistics	
received	The total number of IP packets received by the device.
sent	The total number of IP packets originated and sent by the device.

TABLE 18 CLI display of IP traffic statistics - Layer 3 switch (continued)

Field	Description
forwarded	The total number of IP packets received by the device and forwarded to other devices.
filtered	The total number of IP packets filtered by the device.
fragmented	The total number of IP packets fragmented by this device to accommodate the MTU of this device or of another device.
reassembled	The total number of fragmented IP packets that this device re-assembled.
bad header	The number of IP packets dropped by the device due to a bad packet header.
no route	The number of packets dropped by the device because there was no route.
unknown proto	The number of packets dropped by the device because the value in the Protocol field of the packet header is unrecognized by this device.
no buffer	This information is used by Brocade customer support.
other errors	The number of packets dropped due to error types other than those listed above.
ICMP statistics	
The ICMP statistics are derived from RFC 792, "Internet Control Message Protocol", RFC 950, "Internet Standard Subnetting Procedure", and RFC 1256, "ICMP Router Discovery Messages". Statistics are organized into Sent and Received. The field descriptions below apply to each.	
total	The total number of ICMP messages sent or received by the device.
errors	This information is used by Brocade customer support.
unreachable	The number of Destination Unreachable messages sent or received by the device.
time exceed	The number of Time Exceeded messages sent or received by the device.
parameter	The number of Parameter Problem messages sent or received by the device.
source quench	The number of Source Quench messages sent or received by the device.
redirect	The number of Redirect messages sent or received by the device.
echo	The number of Echo messages sent or received by the device.
echo reply	The number of Echo Reply messages sent or received by the device.
timestamp	The number of Timestamp messages sent or received by the device.
timestamp reply	The number of Timestamp Reply messages sent or received by the device.
addr mask	The number of Address Mask Request messages sent or received by the device.
addr mask reply	The number of Address Mask Replies messages sent or received by the device.
irdp advertisement	The number of ICMP Router Discovery Protocol (IRDP) Advertisement messages sent or received by the device.
irdp solicitation	The number of IRDP Solicitation messages sent or received by the device.
UDP statistics	
received	The number of UDP packets received by the device.
sent	The number of UDP packets sent by the device.
no port	The number of UDP packets dropped because they did not have a valid UDP port number.
input errors	This information is used by Brocade customer support.

TABLE 18 CLI display of IP traffic statistics - Layer 3 switch (continued)

Field	Description
TCP statistics	
The TCP statistics are derived from RFC 793, "Transmission Control Protocol".	
active opens	The number of TCP connections opened by sending a TCP SYN to another device.
passive opens	The number of TCP connections opened by this device in response to connection requests (TCP SYNs) received from other devices.
failed attempts	This information is used by Brocade customer support.
active resets	The number of TCP connections this device reset by sending a TCP RESET message to the device at the other end of the connection.
passive resets	The number of TCP connections this device reset because the device at the other end of the connection sent a TCP RESET message.
input errors	This information is used by Brocade customer support.
in segments	The number of TCP segments received by the device.
out segments	The number of TCP segments sent by the device.
retransmission	The number of segments that this device retransmitted because the retransmission timer for the segment had expired before the device at the other end of the connection had acknowledged receipt of the segment.
RIP statistics	
The RIP statistics are derived from RFC 1058, "Routing Information Protocol".	
requests sent	The number of requests this device has sent to another RIP router for all or part of its RIP routing table.
requests received	The number of requests this device has received from another RIP router for all or part of this device RIP routing table.
responses sent	The number of responses this device has sent to another RIP router request for all or part of this device RIP routing table.
responses received	The number of responses this device has received to requests for all or part of another RIP router routing table.
unrecognized	This information is used by Brocade customer support.
bad version	The number of RIP packets dropped by the device because the RIP version was either invalid or is not supported by this device.
bad addr family	The number of RIP packets dropped because the value in the Address Family Identifier field of the packet header was invalid.
bad req format	The number of RIP request packets this router dropped because the format was bad.
bad metrics	This information is used by Brocade customer support.
bad resp format	The number of responses to RIP request packets dropped because the format was bad.
resp not from rip port	This information is used by Brocade customer support.
resp from loopback	The number of RIP responses received from loopback interfaces.
packets rejected	This information is used by Brocade customer support.

Displaying IP information - Layer 2 switches

You can display the following IP configuration information statistics on Layer 2 switches:

- Global IP settings

- ARP entries
- IP traffic statistics

Displaying global IP configuration information

To display the Layer 2 switch IP address and default gateway, enter the **show ip** command.

```
device# show ip
Switch IP address: 192.168.1.2
Subnet mask: 255.255.255.0
Default router address: 192.168.1.1
TFTP server address: None
Configuration filename: None
Image filename: None
```

Syntax: show ip

This display shows the following information.

TABLE 19 CLI display of global IP configuration information - Layer 2 switch

Field	Description
IP configuration	
Switch IP address	The management IP address configured on the Layer 2 switch. Specify this address for Telnet access or Web management access.
Subnet mask	The subnet mask for the management IP address.
Default router address	The address of the default gateway, if you specified one.
Most recent TFTP access	
TFTP server address	The IP address of the most-recently contacted TFTP server, if the switch has contacted a TFTP server since the last time the software was reloaded or the switch was rebooted.
Configuration filename	The name under which the Layer 2 switch startup-config file was uploaded or downloaded during the most recent TFTP access.
Image filename	The name of the Layer 2 switch flash image (system software file) that was uploaded or downloaded during the most recent TFTP access.

Displaying ARP entries

To display the entries the Layer 2 switch has placed in its ARP cache, enter the **show arp** command from any level of the CLI. This command shows the total number of ARPs for the default VRF instance.

NOTE

To display the ARP maximum capacity for your device, enter the **show default values** command.

```
device# show arp
Total Arp Entries : 1
No.
1      IP      Mac      Port Age  VlanId
      192.168.1.170  0000.0011.d042  7    0    1
```

Displaying IP configuration information and statistics

Syntax: show arp

TABLE 20 CLI display of ARP cache

Syntax: show arp	Description
Field	
Total ARP Entries	The number of entries in the ARP cache.
IP	The IP address of the device.
Mac	The MAC address of the device. NOTE If the MAC address is all zeros, the entry is for the default gateway, but the Layer 2 switch does not have a link to the gateway.
Port	The port on which the entry was learned.
Age	The number of minutes the entry has remained unused. If this value reaches the ARP aging period, the entry is removed from the cache.
VlanId	The VLAN the port that learned the entry is in. NOTE If the MAC address is all zeros, this field shows a random VLAN ID, since the Layer 2 switch does not yet know which port the device for this entry is attached to.

Displaying IP traffic statistics

To display IP traffic statistics on a Layer 2 switch, enter the **show ip traffic** command at any CLI level.

```
device# show ip traffic
IP Statistics
 27 received, 24 sent
 0 fragmented, 0 reassembled, 0 bad header
 0 no route, 0 unknown proto, 0 no buffer, 0 other errors
ICMP Statistics
Received:
 0 total, 0 errors, 0 unreachable, 0 time exceed
 0 parameter, 0 source quench, 0 redirect, 0 echo,
 0 echo reply, 0 timestamp, 0 timestamp rely, 0 addr mask
 0 addr mask reply, 0 irdp advertisement, 0 irdp solicitation
Sent:
 0 total, 0 errors, 0 unreachable, 0 time exceed
 0 parameter, 0 source quench, 0 redirect, 0 echo,
 0 echo reply, 0 timestamp, 0 timestamp rely, 0 addr mask
 0 addr mask reply, 0 irdp advertisement, 0 irdp solicitation
UDP Statistics
 0 received, 0 sent, 0 no port, 0 input errors
TCP Statistics
 1 current active tcbs, 4 tcbs allocated, 0 tcbs freed 0 tcbs protected
 0 active opens, 0 passive opens, 0 failed attempts
 0 active resets, 0 passive resets, 0 input errors
 27 in segments, 24 out segments, 0 retransmission
```

Syntax: show ip traffic

The **show ip traffic** command displays the following information.

TABLE 21 CLI display of IP traffic statistics - Layer 2 switch

Field	Description
IP statistics	

TABLE 21 CLI display of IP traffic statistics - Layer 2 switch (continued)

Field	Description
received	The total number of IP packets received by the device.
sent	The total number of IP packets originated and sent by the device.
fragmented	The total number of IP packets fragmented by this device to accommodate the MTU of this device or of another device.
reassembled	The total number of fragmented IP packets that this device re-assembled.
bad header	The number of IP packets dropped by the device due to a bad packet header.
no route	The number of packets dropped by the device because there was no route.
unknown proto	The number of packets dropped by the device because the value in the Protocol field of the packet header is unrecognized by this device.
no buffer	This information is used by Brocade customer support.
other errors	The number of packets that this device dropped due to error types other than the types listed above.
ICMP statistics	
The ICMP statistics are derived from RFC 792, "Internet Control Message Protocol", RFC 950, "Internet Standard Subnetting Procedure", and RFC 1256, "ICMP Router Discovery Messages". Statistics are organized into Sent and Received. The field descriptions below apply to each.	
total	The total number of ICMP messages sent or received by the device.
errors	This information is used by Brocade customer support.
unreachable	The number of Destination Unreachable messages sent or received by the device.
time exceed	The number of Time Exceeded messages sent or received by the device.
parameter	The number of Parameter Problem messages sent or received by the device.
source quench	The number of Source Quench messages sent or received by the device.
redirect	The number of Redirect messages sent or received by the device.
echo	The number of Echo messages sent or received by the device.
echo reply	The number of Echo Reply messages sent or received by the device.
timestamp	The number of Timestamp messages sent or received by the device.
timestamp reply	The number of Timestamp Reply messages sent or received by the device.
addr mask	The number of Address Mask Request messages sent or received by the device.
addr mask reply	The number of Address Mask Replies messages sent or received by the device.
irdp advertisement	The number of ICMP Router Discovery Protocol (IRDP) Advertisement messages sent or received by the device.
irdp solicitation	The number of IRDP Solicitation messages sent or received by the device.
UDP statistics	
received	The number of UDP packets received by the device.
sent	The number of UDP packets sent by the device.
no port	The number of UDP packets dropped because the packet did not contain a valid UDP port number.
input errors	This information is used by Brocade customer support.
TCP statistics	

TABLE 21 CLI display of IP traffic statistics - Layer 2 switch (continued)

Field	Description
The TCP statistics are derived from RFC 793, "Transmission Control Protocol".	
current active tcbs	The number of TCP Control Blocks (TCBs) that are currently active.
tcbs allocated	The number of TCBs that have been allocated.
tcbs freed	The number of TCBs that have been freed.
tcbs protected	This information is used by Brocade customer support.
active opens	The number of TCP connections opened by this device by sending a TCP SYN to another device.
passive opens	The number of TCP connections opened by this device in response to connection requests (TCP SYNs) received from other devices.
failed attempts	This information is used by Brocade customer support.
active resets	The number of TCP connections this device reset by sending a TCP RESET message to the device at the other end of the connection.
passive resets	The number of TCP connections this device reset because the device at the other end of the connection sent a TCP RESET message.
input errors	This information is used by Brocade customer support.
in segments	The number of TCP segments received by the device.
out segments	The number of TCP segments sent by the device.
retransmission	The number of segments that this device retransmitted because the retransmission timer for the segment had expired before the device at the other end of the connection had acknowledged receipt of the segment.

IPv6 Addressing

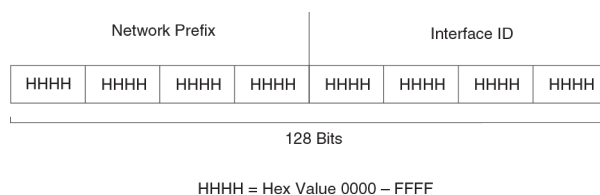
• IPv6 addressing overview.....	125
• Full Layer 3 IPv6 feature support.....	128
• IPv6 CLI command support	128
• IPv6 host address on a Layer 2 switch.....	130
• Configuring the management port for an IPv6 automatic address configuration.....	132
• Configuring basic IPv6 connectivity on a Layer 3 switch.....	132
• IPv6 over IPv4 tunnels.....	136
• IPv6 management (IPv6 host support).....	139
• IPv6 ICMP feature configuration.....	145
• IPv6 neighbor discovery configuration.....	147
• IPv6 neighbor discovery inspection.....	153
• IPv6 MTU.....	156
• Static neighbor entries configuration.....	157
• Limiting the number of hops an IPv6 packet can traverse.....	158
• IPv6 source routing security enhancements.....	158
• TCAM space configuration.....	158
• Displaying global IPv6 information.....	161
• Clearing global IPv6 information.....	172

IPv6 addressing overview

IPv6 was designed to replace IPv4, the Internet protocol that is most commonly used currently throughout the world. IPv6 increases the number of network address bits from 32 (IPv4) to 128 bits, which provides more than enough unique IP addresses to support all of the network devices on the planet into the future. IPv6 is expected to quickly become the network standard.

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

FIGURE 10 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 less cumbersome, you can do the following:

- 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, keep the following in mind:

- 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 10](#), 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 left-most bits of the IPv6 address.

The following is an example of an IPv6 prefix.

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

IPv6 address types

As with IPv4 addresses, you can assign multiple IPv6 addresses to a switch interface. [IPv6 address types](#) presents the three major types of IPv6 addresses that you can assign to a switch interface.

A major difference between IPv4 and IPv6 addresses is that IPv6 addresses support scope, which describes the topology in which the address may be used as a unique identifier for an interface or set of interfaces.

Unicast and multicast addresses support scoping as follows:

- Unicast addresses support two types of scope: global scope and local scope. In turn, local scope supports site-local addresses and link-local addresses. [IPv6 address types](#) describes global, site-local, and link-local addresses and the topologies in which they are used.
- Multicast addresses support a scope field, which [IPv6 address types](#) describes.

TABLE 22 IPv6 address types

Address type	Description	Address structure
Unicast	An address for a single interface. A packet sent to a unicast address is delivered to the interface identified by the address.	<p>Depends on the type of the unicast address:</p> <ul style="list-style-type: none"> • Aggregatable global address--An address equivalent to a global or public IPv4 address. The address structure is as follows: a fixed prefix of 2000::/3 (001), a 45-bit global routing prefix, a 16-bit subnet ID, and a 64-bit interface ID. • Site-local address--An address used within a site or intranet. (This address is similar to a private IPv4 address.) A site consists of multiple network links. The address structure is as follows: a fixed prefix of FEC0::/10 (1111 1110 11), a 16-bit subnet ID, and a 64-bit interface ID. • Link-local address--An address used between directly connected nodes on a single network link. The address structure is as follows: a fixed prefix of FE80::/10 (1111 1110 10) and a 64-bit interface ID. • IPv4-compatible address--An address used in IPv6 transition mechanisms that tunnel IPv6 packets

TABLE 22 IPv6 address types (continued)

Address type	Description	Address structure
		<p>dynamically over IPv4 infrastructures. The address embeds an IPv4 address in the low-order 32 bits and the high-order 96 bits are zeros. The address structure is as follows: 0:0:0:0:0:A.B.C.D.</p> <ul style="list-style-type: none"> • Loopback address--An address (0:0:0:0:0:0:0:1 or ::1) that a switch can use to send an IPv6 packet to itself. You cannot assign a loopback address to a physical interface. • Unspecified address--An address (0:0:0:0:0:0:0:0 or ::) that a node can use until you configure an IPv6 address for it.
Multicast	An address for a set of interfaces belonging to different nodes. Sending a packet to a multicast address results in the delivery of the packet to all interfaces in the set.	A multicast address has a fixed prefix of FF00::/8 (1111 1111). The next 4 bits define the address as a permanent or temporary address. The next 4 bits define the scope of the address (node, link, site, organization, global).
Anycast	An address for a set of interfaces belonging to different nodes. Sending a packet to an anycast address results in the delivery of the packet to the closest interface identified by the address.	<p>An anycast address looks similar to a unicast address, because it is allocated from the unicast address space. If you assign a unicast address to multiple interfaces, it is an anycast address. An interface assigned an anycast address must be configured to recognize the address as an anycast address.</p> <p>An anycast address can be assigned to a switch only.</p> <p>An anycast address must not be used as the source address of an IPv6 packet.</p>

A switch automatically configures a link-local unicast address for an interface by using the prefix of FE80::/10 (1111 1110 10) and a 64-bit interface ID. The 128-bit IPv6 address is then subjected to duplicate address detection to ensure that the address is unique on the link. If desired, you can override this automatically configured address by explicitly configuring an address.

NOTE

Brocade FastIron devices support RFC 2526, which requires that within each subnet, the highest 128 interface identifier values reserved for assignment as subnet anycast addresses. Thus, if you assign individual IPv6 addresses within a subnet, the second highest IPv6 address in the subnet does not work.

IPv6 stateless auto-configuration

Brocade routers use the IPv6 stateless autoconfiguration feature to enable a host on a local link to automatically configure its interfaces with new and globally unique IPv6 addresses associated with its location. The automatic configuration of a host interface is performed without the use of a server, such as a Dynamic Host Configuration Protocol (DHCP) server, or manual configuration.

The automatic configuration of a host interface works in the following way: a switch on a local link periodically sends switch advertisement messages containing network-type information, such as the 64-bit prefix of the local link and the default route, to all nodes on the link. When a host on the link receives the message, it takes the local link prefix from the message and appends a 64-bit

interface ID, thereby automatically configuring its interface. (The 64-bit interface ID is derived from the MAC address of the host's NIC.) The 128-bit IPv6 address is then subjected to duplicate address detection to ensure that the address is unique on the link.

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 uses neighbor solicitation messages to verify that a unicast IPv6 address is unique.

NOTE

For the stateless auto configuration feature to work properly, the advertised prefix length in switch advertisement messages must always be 64 bits.

The IPv6 stateless autoconfiguration feature can also automatically reconfigure a host's interfaces if you change the ISP for the host's network. (The host's interfaces must be renumbered with the IPv6 prefix of the new ISP.)

The renumbering occurs in the following way: a switch on a local link periodically sends advertisements updated with the prefix of the new ISP to all nodes on the link. (The advertisements still contain the prefix of the old ISP.) A host can use the addresses created from the new prefix and the existing addresses created from the old prefix on the link. When you are ready for the host to use the new addresses only, you can configure the lifetime parameters appropriately using the **ipv6 nd prefix-advertisement** command. During this transition, the old prefix is removed from the switch advertisements. At this point, only addresses that contain the new prefix are used on the link.

Full Layer 3 IPv6 feature support

The following IPv6 Layer 3 features are supported only with the IPv6 Layer 3 PROM, Software-based Licensing, IPv6-series hardware, and the full Layer 3 image:

- OSPF V3
- RIPng
- IPv6 ICMP redirect messages
- IPv6 route redistribution
- IPv6 over IPv4 tunnels in hardware
- IPv6 Layer 3 forwarding
- BGP4+
- IPv6 Multicast routing
- DHCPv6 Relay Agent

NOTE

IPv6 static routes and IPv6 unicast routing (multicast routing is not supported) are not supported in the base Layer 3 software images.

IPv6 CLI command support

[IPv6 CLI command support](#) lists the IPv6 CLI commands supported.

TABLE 23 IPv6 CLI command support

IPv6 command	Description	Switch code	Router code
clear ipv6 cache	Deletes all entries in the dynamic host cache.		X

TABLE 23 IPv6 CLI command support (continued)

IPv6 command	Description	Switch code	Router code
clear ipv6 mld-snooping	Deletes MLD-snooping-related counters or cache entries.	X	X
clear ipv6 neighbor	Deletes all dynamic entries in the IPv6 neighbor table.	X	X
clear ipv6 ospf	Clears OSPF-related entries.		X
clear ipv6 rip	Clears RIP-related entries.		X
clear ipv6 route	Deletes all dynamic entries in the IPv6 route table.		X
clear ipv6 traffic	Resets all IPv6 packet counters.	X	X
clear ipv6 tunnel	Clears statistics for IPv6 tunnels		X
copy tftp	Downloads a copy of a Brocade software image from a TFTP server into the system flash using IPv6.	X	X
debug ipv6	Displays IPv6 debug information.	X	X
ipv6 access-class	Configures access control for IPv6 management traffic.	X	X
ipv6 access-list	Configures an IPv6 access control list for IPv6 access control.	X	X
ipv6 address	Configures an IPv6 address on an interface (router) or globally (switch)	X	X
ipv6 debug	Enables IPv6 debugging.	X	X
ipv6 dns domain-name	Configures an IPv6 domain name.	X	X
ipv6 dns server-address	Configures an IPv6 DNS server address.	X	X
ipv6 enable	Enables IPv6 on an interface.	X	X
ipv6 hop-limit	Sets the IPv6 hop limit.		X
ipv6 icmp	Configures IPv6 ICMP parameters		X
ipv6 load-sharing	Enables IPv6 load sharing		X
ipv6 mld-snooping	Configures MLD snooping	X	X
ipv6 mtu	Configures the maximum length of an IPv6 packet that can be transmitted on a particular interface.		X
ipv6 nd	Configures neighbor discovery.		X
ipv6 neighbor	Maps a static IPv6 address to a MAC address in the IPv6 neighbor table.		X
ipv6 ospf	Configures OSPF V3 parameters on an interface.		X
ipv6 prefix-list	Builds an IPv6 prefix list.		X
ipv6 redirects	Enables the sending of ICMP redirect messages on an interface.		X
ipv6 rip	Configures RIPng parameters on an interface		X
ipv6 route	Configures an IPv6 static route.		X
ipv6 router	Enables an IPv6 routing protocol.		X
ipv6 traffic-filter	Applies an IPv6 ACL to an interface.	X	X

TABLE 23 IPv6 CLI command support (continued)

IPv6 command	Description	Switch code	Router code
ipv6 unicast-routing	Enables IPv6 unicast routing.		X
log host ipv6	Configures the IPv6 Syslog server.	X	X
ping ipv6	Performs an ICMP for IPv6 echo test.	X	X
show ipv6	Displays some global IPv6 parameters, such as IPv6 DNS server address.	X	X
show ipv6 access-list	Displays configured IPv6 access control lists.	X	X
show ipv6 cache	Displays the IPv6 host cache.		X
show ipv6 interface	Displays IPv6 information for an interface.		X
show ipv6 mld-snooping	Displays information about MLD snooping.	X	X
show ipv6 neighbor	Displays the IPv6 neighbor table.	X	X
show ipv6 ospf	Displays information about OSPF V3.		X
show ipv6 prefix-lists	Displays the configured IPv6 prefix lists.		X
show ipv6 rip	Displays information about RIPng.		X
show ipv6 route	Displays IPv6 routes.		X
show ipv6 router	Displays IPv6 local routers.		X
show ipv6 tcp	Displays information about IPv6 TCP sessions.	X	X
show ipv6 traffic	Displays IPv6 packet counters.	X	X
show ipv6 tunnel	Displays information about IPv6 tunnels.	X	X
snmp-client ipv6	Restricts SNMP access to a certain IPv6 node.	X	X
snmp-server host ipv6	Specifies the recipient of SNMP notifications.	X	X
telnet	Enables a Telnet connection from the Brocade device to a remote IPv6 host using the console.	X	X
traceroute ipv6	Traces a path from the Brocade device to an IPv6 host.	X	X
web access-group ipv6	Restricts Web management access to certain IPv6 hosts as determined by IPv6 ACLs.	X	X
web client ipv6	Restricts Web management access to certain IPv6 hosts.	X	X

IPv6 host address on a Layer 2 switch

In a Layer 3 (router) configuration, each port can be configured separately with an IPv6 address. This is accomplished using the interface configuration process that is described in [IPv6 configuration on each router interface](#) on page 132.

There is support for configuring an IPv6 address on the management port as described in [Configuring the management port for an IPv6 automatic address configuration](#) on page 132, and for configuring a system-wide IPv6 address on a Layer 2 switch. Configuration of the system-wide IPv6 address is exactly like configuration of an IPv6 address in router mode, except that the IPv6 configuration is at the Global CONFIG level instead of at the Interface level.

The process for defining the system-wide interface for IPv6 is described in the following sections:

- [Configuring a global or site-local IPv6 address with a manually configured interface ID](#) on page 131
- [Configuring a link-local IPv6 address as a system-wide address for a switch](#) on page 131

NOTE

When configuring an IPv6 host address on a Layer 2 switch that has multiple VLANs, make sure the configuration includes a designated management VLAN that identifies the VLAN to which the global IP address belongs. Refer to "Designated VLAN for Telnet management sessions to a Layer 2 Switch" section in the *Brocade FastIron Security Configuration Guide*.

Configuring a global or site-local IPv6 address with a manually configured interface ID

To configure a global or site-local IPv6 address with a manually-configured interface ID, such as a system-wide address for a switch, enter a command similar to the following at the Global CONFIG level.

```
device(config)# ipv6 address 2001:DB8:12D:1300:240:D0FF:FE48:4000:1/64
```

Syntax: `ipv6 address ipv6-prefix/prefix-length`

You must specify the `ipv6-prefix` parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the `prefix-length` parameter in decimal value. A slash mark (/) must follow the `ipv6-prefix` parameter and precede the `prefix-length` parameter.

Configuring a link-local IPv6 address as a system-wide address for a switch

To enable IPv6 and automatically configure a global interface enter commands such as the following.

```
device(config)#ipv6 enable
```

This command enables IPv6 on the switch and specifies that the interface is assigned an automatically computed link-local address.

Syntax: `[no] ipv6 enable`

To override a link-local address that is automatically computed for the global interface with a manually configured address, enter a command such as the following.

```
device(config)#ipv6 address FE80::240:D0FF:FE48:4672 link-local
```

This command explicitly configures the link-local address FE80::240:D0FF:FE48:4672 for the global interface.

Syntax: `ipv6 address ipv6-address link-local`

You must specify the `ipv6-address` parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **link-local** keyword indicates that the router interface should use the manually configured link-local address instead of the automatically computed link-local address.

Configuring the management port for an IPv6 automatic address configuration

You can have the management port configured to automatically obtain an IPv6 address for your Brocade device. This process is the same for any other port and is described in detail in the section [Configuring a global or site-local IPv6 address on an interface](#) on page 133.

Configuring basic IPv6 connectivity on a Layer 3 switch

To configure basic IPv6 connectivity on a Brocade Layer 3 Switch, you must:

- Enable IPv6 routing globally on the switch
- Configure an IPv6 address or specifically enable IPv6 on each router interface over which you plan to forward the IPv6 traffic
- Configure IPv4 and IPv6 protocol stacks. (This task is mandatory only if you want a router interface to send and receive both IPv4 and IPv6 traffic.)

All other configuration tasks in this chapter are optional.

NOTE

- When you configure an IPv6 address on a device, a syslog appears stating that the IPv6 address has been added. You cannot configure the same IPv6 address on the device again.
- When you attempt to configure the same IPv6 address on the device, this syslog appears: "Error: duplicate IPv6 address!".
- When you configure a different IPv6 address on the device, two different syslogs appear stating that the existing IPv6 address has been removed and the new IPv6 address has been configured.

Enabling IPv6 routing

By default, IPv6 routing is disabled. To enable the forwarding of IPv6 traffic globally on the Layer 3 switch, enter the following command.

```
device(config)#ipv6 unicast-routing
```

Syntax: [no] ipv6 unicast-routing

To disable the forwarding of IPv6 traffic globally on the Brocade device, enter the **no** form of this command.

IPv6 configuration on each router interface

To forward IPv6 traffic on a router interface, the interface must have an IPv6 address, or IPv6 must be explicitly enabled. By default, an IPv6 address is not configured on a router interface.

If you choose to configure a global or site-local IPv6 address for an interface, IPv6 is also enabled on the interface. Further, when you configure a global or site-local IPv6 address, you must decide on one of the following in the low-order 64 bits:

- A manually configured interface ID.
- An automatically computed EUI-64 interface ID.

If you prefer to assign a link-local IPv6 address to the interface, you must explicitly enable IPv6 on the interface, which causes a link-local address to be automatically computed for the interface. If preferred, you can override the automatically configured link-local address with an address that you manually configure.

This section provides the following information:

- Configuring a global or site-local address with a manually configured or automatically computed interface ID for an interface.
- Automatically or manually configuring a link-local address for an interface.
- Configuring IPv6 anycast addresses

Configuring a global or site-local IPv6 address on an interface

Configuring a global or site-local IPv6 address on an interface does the following:

- Automatically configures an interface ID (a link-local address), if specified.
- Enables IPv6 on that interface.

Additionally, the configured interface automatically joins the following required multicast groups for that link:

- Solicited-node multicast group FF02::0:0:0:1:FF00::/104 for each unicast address assigned to the interface.
- Solicited-node for subnet anycast address for each unicast assigned address
- Solicited-node for anycast address FF02::0:0:0:1:FF00::0000
- All-nodes link-local multicast group FF02::1
- All-routers link-local multicast group FF02::2

The neighbor discovery feature sends messages to these multicast groups. For more information, refer to [IPv6 neighbor discovery configuration](#) on page 147.

Configuring a global or site-local IPv6 address with a manually configured interface ID

To configure a global or site-local IPv6 address, including a manually configured interface ID, for an interface, enter commands such as the following.

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

These commands configure the global prefix 2001:DB8:12d:1300::/64 and the interface ID ::240:D0FF:FE48:4672, and enable IPv6 on Ethernet interface 1/3/1.

Syntax: **ipv6 address** *ipv6-prefix/prefix-length*

You must specify the *ipv6-prefix* parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

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.

To configure a /122 address on a VE enter commands similar to the following.

```
device(config-vlan-11)#int ve11
device(config-vif-11)#ipv6 add 2001:DB8::1/122
device(config-vif-11)#sh ipv6 int
Routing Protocols : R - RIP  O - OSPF
Interface      Status      Routing  Global Unicast Address
VE 11          up/up          2001:DB8::1/122
device(config-vif-11)#sh ipv6 route
IPv6 Routing Table - 1 entries:
Type Codes:  C - Connected, S - Static, R - RIP, O - OSPF, B - BGP
OSPF Sub Type Codes:  O - Intra, Oi - Inter, O1 - Type1 external, O2 - Type2 external
```

Configuring basic IPv6 connectivity on a Layer 3 switch

Type	IPv6 Prefix	Next Hop Router	Interface	Dis/Metric
C	2001:DB8::/122	::	ve 11	0/0

Configuring a global IPv6 address with an automatically computed EUI-64 interface ID

To configure a global IPv6 address with an automatically computed EUI-64 interface ID in the low-order 64-bits, enter commands such as the following.

```
device(config)#interface ethernet 1/3/1
device(config-if-e1000-1/3/1)#ipv6 address 2001:DB8:12D:1300::/64 eui-64
```

These commands configure the global prefix 2001:DB8:12d:1300::/64 and an interface ID, and enable IPv6 on Ethernet interface 1/3/1.

Syntax: `ipv6 address ipv6-prefix/prefix-length eui-64`

You must specify the `ipv6-prefix` parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

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.

The **eui-64** keyword configures the global address with an EUI-64 interface ID in the low-order 64 bits. The interface ID is automatically constructed in IEEE EUI-64 format using the interface's MAC address.

Configuring a link-local IPv6 address on an interface

To explicitly enable IPv6 on a router interface without configuring a global or site-local address for the interface, enter commands such as the following.

```
device(config)#interface ethernet 1/3/1
device(config-if-e1000-1/3/1)#ipv6 enable
```

These commands enable IPv6 on Ethernet interface 1/3/1 and specify that the interface is assigned an automatically computed link-local address.

Syntax: `[no] ipv6 enable`

NOTE

When configuring VLANs that share a common tagged interface with a physical or Virtual Ethernet (VE) interface, Brocade recommends that you override the automatically computed link-local address with a manually configured unique address for the interface. If the interface uses the automatically computed address, which in the case of physical and VE interfaces is derived from a global MAC address, all physical and VE interfaces will have the same MAC address.

To override a link-local address that is automatically computed for an interface with a manually configured address, enter commands such as the following.

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

These commands explicitly configure the link-local address FE80::240:D0FF:FE48:4672 for Ethernet interface 1/3/1.

Syntax: `ipv6 address ipv6-address link-local`

You must specify the `ipv6-address` parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **link-local** keyword indicates that the router interface should use the manually configured link-local address instead of the automatically computed link-local address.

Configuring an IPv6 anycast address on an interface

In IPv6, an anycast address is an address for a set of interfaces belonging to different nodes. Sending a packet to an anycast address results in the delivery of the packet to the closest interface configured with the anycast address.

An anycast address looks similar to a unicast address, because it is allocated from the unicast address space. If you assign an IPv6 unicast address to multiple interfaces, it is an anycast address. On the Brocade device, you configure an interface assigned an anycast address to recognize the address as an anycast address.

For example, the following commands configure an anycast address on interface 1/2/1.

```
device(config)#int e 1/2/1
device(config-if-e1000-1/2/1)#ipv6 address 2001:DB8::/64 anycast
```

Syntax: `ipv6 address ipv6-prefix/prefix-length [anycast]`

IPv6 anycast addresses are described in detail in RFC 1884. Refer to RFC 2461 for a description of how the IPv6 Neighbor Discovery mechanism handles anycast addresses.

Configuring IPv4 and IPv6 protocol stacks

One situation in which you must configure a router to run both IPv4 and IPv6 protocol stacks is if it is deployed as an endpoint for an IPv6 over IPv4 tunnel.

Each router interface that will send and receive both IPv4 and IPv6 traffic must be configured with an IPv4 address and an IPv6 address. (An alternative to configuring a router interface with an IPv6 address is to explicitly enable IPv6 using the **ipv6 enable** command. For more information about using this command, refer to [Configuring a link-local IPv6 address on an interface](#) on page 134.)

To configure a router interface to support both the IPv4 and IPv6 protocol stacks, use commands such as the following.

```
device(config)#ipv6 unicast-routing
device(config)#interface ethernet 1/3/1
device(config-if-e1000-1/3/1)#ip address 10.168.1.1 255.255.255.0
device(config-if-e1000-1/3/1)#ipv6 address 2001:DB8:12d:1300::/64 eui-64
```

These commands globally enable IPv6 routing and configure an IPv4 address and an IPv6 address for Ethernet interface 1/3/1.

Syntax: `[no] ipv6 unicast-routing`

To disable IPv6 traffic globally on the router, enter the **no** form of this command.

Syntax: `ip address ip-address sub-net-mask [secondary]`

You must specify the ip-address parameter using 8-bit values in dotted decimal notation.

You can specify the sub-net-mask parameter in either dotted decimal notation or as a decimal value preceded by a slash mark (/).

The **secondary** keyword specifies that the configured address is a secondary IPv4 address.

To remove the IPv4 address from the interface, enter the **no** form of this command.

Syntax: `ipv6 address ipv6-prefix /prefix-length [eui-64]`

This syntax specifies a global or site-local IPv6 address. For information about configuring a link-local IPv6 address, refer to [Configuring a link-local IPv6 address on an interface](#) on page 134.

You must specify the ipv6-prefix parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

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.

The **eui-64** keyword configures the global address with an EUI-64 interface ID in the low-order 64 bits. The interface ID is automatically constructed in IEEE EUI-64 format using the interface's MAC address. If you do not specify the **eui-64** keyword, you must manually

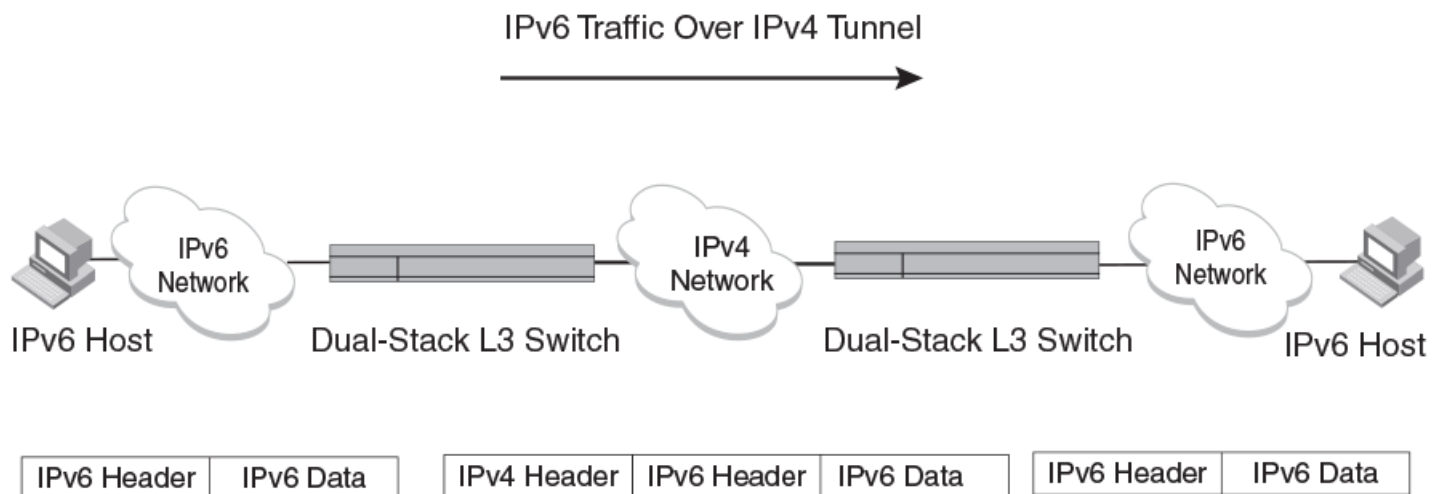
configure the 64-bit interface ID as well as the 64-bit network prefix. For more information about manually configuring an interface ID, refer to [Configuring a global or site-local IPv6 address on an interface](#) on page 133.

IPv6 over IPv4 tunnels

To enable communication between isolated IPv6 domains using the IPv4 infrastructure, you can manually configure IPv6 over IPv4 tunnels that provide static point-point connectivity.

As shown in the following illustration, these tunnels encapsulate an IPv6 packet within an IPv4 packet.

FIGURE 11 IPv6 over an IPv4 tunnel



A manually configured tunnel establishes a permanent link between switches in IPv6 domains. A manually configured tunnel has explicitly configured IPv4 addresses for the tunnel source and destination.

This tunneling mechanism requires that the Layer 3 switch at each end of the tunnel run both IPv4 and IPv6 protocol stacks. The Layer 3 switches running both protocol stacks, or dual-stack routers, can interoperate directly with both IPv4 and IPv6 end systems and routers. Refer to the "Configuring IPv4 and IPv6 protocol stacks" section in the *Brocade FastIron Layer 3 Routing Configuration Guide*.

IPv6 over IPv4 tunnel configuration notes

- The local tunnel configuration must include both source and destination addresses.
- The remote side of the tunnel must have the opposite source/destination pair.
- A tunnel interface supports static and dynamic IPv6 configuration settings and routing protocols.
- Duplicate Address Detection (DAD) is not currently supported with IPv6 tunnels. Make sure tunnel endpoints do not have duplicate IP addresses.
- Neighbor Discovery (ND) is not supported with IPv6 tunnels.
- If a tunnel source port is a multi-homed IPv4 source, the tunnel will use the first IPv4 address only. For proper tunnel operation, use the **ip address** option.

Configuring a manual IPv6 tunnel

You can use a manually configured tunnel to connect two isolated IPv6 domains. You should deploy this point-to-point tunneling mechanism if you need a permanent and stable connection.

To configure a manual IPv6 tunnel, enter commands such as the following on a Layer 3 Switch running both IPv4 and IPv6 protocol stacks on each end of the tunnel.

```
device(config)#interface tunnel 1
device(config-tnif-1)#tunnel source ethernet 1/3/1
device(config-tnif-1)#tunnel destination 10.162.100.1
device(config-tnif-1)#tunnel mode ipv6ip
device(config-tnif-1)#ipv6 enable
```

This example creates tunnel interface 1 and assigns a link local IPv6 address with an automatically computed EUI-64 interface ID to it. The IPv4 address assigned to Ethernet interface 1/3/1 is used as the tunnel source, while the IPv4 address 10.168.100.1 is configured as the tunnel destination. The tunnel mode is specified as a manual IPv6 tunnel. Finally, the tunnel is enabled. Note that instead of entering **ipv6 enable**, you could specify an IPv6 address, for example, **ipv6 address 2001:DB8:384d:34::/64 eui-64**, which would also enable the tunnel.

Syntax: **[no] interface***tunnel number*

For the *number* parameter, specify a value between 1-8.

Syntax: **[no] tunnel***source ipv4-address | ethernet port | loopback number | ve number*

The tunnel source can be an IP address or an interface.

For *ipv4-address*, use 8-bit values in dotted decimal notation.

The **ethernet | loopback | ve** parameter specifies an interface as the tunnel source. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a loopback, VE, or interface, also specify the loopback, VE, or number, respectively.

Syntax: **[no] tunnel***destination ipv4-address*

Specify the *ipv4-address* parameter using 8-bit values in dotted decimal notation.

Syntax: **[no] tunnel***mode ipv6ip*

ipv6ip indicates that this is an IPv6 manual tunnel.

Syntax: **ipv6 enable**

The **ipv6 enable** command enables the tunnel. Alternatively, you could specify an IPv6 address, which would also enable the tunnel.

Syntax: **ipv6 address** *ipv6-prefix / prefix-length [eui-64]*

The **ipv6 address** command enables the tunnel. Alternatively, you could enter **ipv6 enable**, which would also enable the tunnel.

Specify the *ipv6-prefix* parameter in hexadecimal format using 16-bit values between colons as documented in RFC 2373.

Specify the *prefix-length* parameter as a decimal value. A slash mark (/) must follow the *ipv6-prefix* parameter and precede the *prefix-length* parameter. The **eui-64** keyword configures the global address with an EUI-64 interface ID in the low-order 64 bits. The interface ID is automatically constructed in IEEE EUI-64 format using the interface's MAC address.

Clearing IPv6 tunnel statistics

You can clear statistics (reset all fields to zero) for all IPv6 tunnels or for a specific tunnel interface.

For example, to clear statistics for tunnel 1, enter the following command at the Privileged EXEC level or any of the Config levels of the CLI.

```
device#clear ipv6 tunnel 1
```

To clear statistics for all IPv6 tunnels, enter the following command.

```
device#clear ipv6 tunnel
```

Syntax: `clear ipv6 tunnel [number]`

The *number* parameter specifies the tunnel number.

Displaying IPv6 tunnel information

Use the commands in this section to display the configuration, status, and counters associated with IPv6 tunnels.

Displaying a summary of tunnel information

To display a summary of tunnel information, enter the following command at any level of the CLI.

```
device#show ipv6 tunnel
IPv6 Tunnels
  Tunnel  Mode      Packet Received  Packet Sent
  1        configured    0                0
  2        configured    0              22419
```

Syntax: `show ipv6tunnel`

This display shows the following information.

TABLE 24 IPv6 tunnel summary information

Field	Description
Tunnel	The tunnel interface number.
Mode	The tunnel mode. Possible modes include the following: <ul style="list-style-type: none"> configured - Indicates a manually configured tunnel.
Packet Received	The number of packets received by a tunnel interface. Note that this is the number of packets received by the CPU. It does not include the number of packets processed in hardware.
Packet Sent	The number of packets sent by a tunnel interface. Note that this is the number of packets sent by the CPU. It does not include the number of packets processed in hardware.

Displaying interface level IPv6 settings

To display Interface level IPv6 settings for tunnel interface 1, enter the following command at any level of the CLI.

```
device#show ipv6 inter tunnel 1
Interface Tunnel 1 is up, line protocol is up
IPv6 is enabled, link-local address is fe80::3:4:2 [Preferred]
Global unicast address(es):
  1001::1 [Preferred], subnet is 1001::/64
  1011::1 [Preferred], subnet is 1011::/64
Joined group address(es):
  ff02::1:ff04:2
  ff02::5
  ff02::1:ff00:1
```

```

ff02::2
ff02::1
MTU is 1480 bytes
ICMP redirects are enabled
No Inbound Access List Set
No Outbound Access List Set
OSPF enabled

```

The display command above reflects the following configuration.

```

device#show running-config interface tunnel 1
!
interface tunnel 1
port-name ManualTunnel1
tunnel mode ipv6ip
tunnel source loopback 1
tunnel destination 10.1.1.1
ipv6 address 1011::1/64
ipv6 address 1001::1/64
ipv6 ospf area 0

```

TABLE 25 Interface level IPv6 tunnel information

Field	Description
Interface Tunnel status	<p>The status of the tunnel interface can be one of the following:</p> <ul style="list-style-type: none"> up - IPv4 connectivity is established. down - The tunnel mode is not set. administratively down - The tunnel interface was disabled with the disable command.
Line protocol status	<p>The status of the line protocol can be one of the following:</p> <ul style="list-style-type: none"> up - IPv6 is enabled through the ipv6 enable or ipv6 address command. down - The line protocol is not functioning and is down.

IPv6 management (IPv6 host support)

You can configure a Brocade switch to serve as an IPv6 host in an IPv6 network. An IPv6 host has IPv6 addresses on its interfaces, but does not have full IPv6 routing enabled on it.

Configuring IPv6 management ACLs

When you enter the `ipv6 access-list` command, the Brocade 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.

```

FastIron(config)#ipv6 access-list netw
FastIron(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.

Restricting SNMP access to an IPv6 node

You can restrict SNMP access to the device to the IPv6 host whose IP address you specify. To do so, enter a command such as the following.

```
device(config)#snmp-client ipv6 2001:DB8:89::23
```

Syntax: `snmp-client ipv6 ipv6-address`

The ipv6-address you specify must be in hexadecimal format using 16-bit values between colons as documented in RFC 2373.

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 will go to the same SNMP trap receiver or set of receivers, typically one or more host devices on the network. To do so, enter a command such as the following.

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

Syntax: `snmp-server host ipv6 ipv6-address`

The ipv6-address you specify must be in hexadecimal format using 16-bit values between colons as documented in RFC 2373.

Configuring SNMP V3 over IPv6

Brocade devices support IPv6 for SNMP version 3. For more information about how to configure SNMP, refer to *Brocade FastIron Management Configuration Guide*.

Secure Shell, SCP, and IPv6

Secure Shell (SSH) is a mechanism that allows secure remote access to management functions on the Brocade device. SSH provides a function similar to Telnet. You can log in to and configure the Brocade 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 Brocade device.

To open an SSH session between an IPv6 host running an SSH client program and the Brocade device, open the SSH client program and specify the IPv6 address of the device. For more information about configuring SSH on the Brocade device, refer to "SSH2 and SCP" chapter in the *Brocade FastIron Security Configuration Guide*.

IPv6 Telnet

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

The **telnet** command establishes a Telnet connection from a Brocade 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. To see the number of open Telnet sessions at any time, enter the **show telnet** command.

To establish a Telnet connection to a remote host with the IPv6 address of 2001:DB8:3de2:c37::6, enter the following command.

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

Syntax: `telnet ipv6-address [port-number | outgoing-interface ethernet port | ve number]`

The `ipv6-address` parameter specifies the address of a remote host. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The `port-number` parameter specifies the port number on which the Brocade device establishes the Telnet connection. You can specify a value between 1 - 65535. If you do not specify a port number, the Brocade device establishes the Telnet connection on port 23.

If the IPv6 address you specify is a link-local address, you must specify the **outgoing-interface** ethernet port | ve number parameter. This parameter identifies the interface that must be used to reach the remote host. If you specify an Ethernet interface, you must also specify the port number associated with the interface. If you specify a VE interface, also specify the VE number.

Establishing a Telnet session from an IPv6 host

To establish a Telnet session from an IPv6 host to the Brocade device, open your Telnet application and specify the IPv6 address of the Layer 3 Switch.

IPv6 traceroute

NOTE

This section describes the `IPv6 traceroute` command. For details about `IPv4 traceroute`, refer to the *Brocade FastIron Monitoring Configuration Guide*.

The **traceroute** command allows you to trace a path from the Brocade device to an IPv6 host.

The CLI displays trace route information for each hop as soon as the information is received. 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 Brocade device displays up to three responses.

For example, to trace the path from the Brocade device to a host with an IPv6 address of 2001:DB8:349e:a384::34, enter the following command:

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

Syntax: `traceroute ipv6 ipv6-address`

The `ipv6-address` parameter specifies the address of a host. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

IPv6 Web management using HTTP and HTTPS

When you have an IPv6 management station connected to a switch with an IPv6 address applied to the management port, you can manage the switch from a Web browser by entering one of the following in the browser address field.

`http://[<ipv6 address>]`

or

`https://[<ipv6 address>]`

NOTE

You must enclose the IPv6 address with square brackets [] in order for the Web browser to work.

Restricting Web management access

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

Restricting Web management access by specifying an IPv6 ACL

You can specify an IPv6 ACL that restricts Web management access to management functions on the device that is acting as the IPv6 host.

Example

```
Brocade(config)# access-list 12 deny host 2000:2383:e0bb::2/128 log
Brocade(config)# access-list 12 deny 30ff:3782::ff89/128 log
Brocade(config)# access-list 12 deny 3000:4828::fe19/128 log
Brocade(config)# access-list 12 permit any
Brocade(config)# web access-group ipv6 12
```

Syntax: `web access-group ipv6 ipv6-ACL-name`

where *ipv6-ACL-name* is a valid IPv6 ACL.

Restricting Web management access to an IPv6 host

You can restrict Web management access to the device to the IPv6 host whose IP address you specify. No other device except the one with the specified IPv6 address can access the Web Management Interface.

Example

```
Brocade(config)#web client ipv6 3000:2383:e0bb::2/128
```

Syntax: `web client ipv6 ipv6-address`

the *ipv6-address* you specify must be in hexadecimal format using 16-bit values between colons as documented in RFC 2373.

Configuring name-to-IPv6 address resolution using IPv6 DNS resolver

The Domain Name Server (DNS) resolver feature lets you use a host name to perform Telnet and ping commands. You can also define a DNS domain on a Brocade device and thereby recognize all hosts within that domain. After you define a domain name, the Brocade device automatically appends the appropriate domain to the host and forwards it to the domain name server.

For example, if the domain "newyork.com" is defined on a Brocade device, and you want to initiate a ping to host "NYC01" on that domain, you need to reference only the host name in the command instead of the host name and its domain name. For example, you could enter either of the following commands to initiate the ping.

```
device#ping ipv6 nyc01
device#ping ipv6 nyc01.newyork.com
```

Defining an IPv6 DNS entry

IPv6 defines new DNS record types to resolve queries for domain names to IPv6 addresses, as well as IPv6 addresses to domain names. Brocade devices running IPv6 software support AAAA DNS records, which are defined in RFC 1886.

AAAA DNS records are analogous to the A DNS records used with IPv4. They store a complete IPv6 address in each record. AAAA records have a type value of 28.

To define an IPv6 DNS server address, enter command such as the following:

```
device(config)#ipv6 dns server-address 2001:DB8::1
```

Syntax: [no] **ipv6 dns server-address** *ipv6-addr* [*ipv6-addr*] [*ipv6-addr*] [*ipv6-addr*]

The *ipv6 dns server-address* parameter sets IPv6 DNS server addresses.

As an example, in a configuration where ftp6.companynet.com is a server with an IPv6 protocol stack, when a user pings ftp6.companynet.com, the Brocade device attempts to resolve the AAAA DNS record. In addition, if the DNS server does not have an IPv6 address, as long as it is able to resolve AAAA records, it can still respond to DNS queries.

Pinging an IPv6 address

The **ping** command allows you to verify the connectivity from a Brocade device to an IPv6 device by performing an ICMP for IPv6 echo test.

For example, to ping a device with the IPv6 address of 2001:DB8:847f:a385:34dd::45 from the Brocade device, enter the following command.

```
device#ping ipv6 2001:DB8:847f:a385:34dd::45
```

Syntax: **ping ipv6** *ipv6-address* [**outgoing-interface** [*port* | *venumber*]] [**source** *ipv6-address*] [**count** *number*] [**timeout** *milliseconds*] [**ttl** *number*] [**size** *bytes*] [**quiet**] [**numeric**] [**no-fragment**] [**verify**] [**data** *1-to-4-byte-hex*] [**brief**]

- The *ipv6-address* parameter specifies the address of the router. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.
- The **outgoing-interface** keyword specifies a physical interface over which you can verify connectivity. If you specify a physical interface, such as an Ethernet interface, you must also specify the port number of the interface. If you specify a virtual interface, such as a VE, you must specify the number associated with the VE.
- The **source** *ipv6-address* parameter specifies an IPv6 address to be used as the origin of the ping packets.
- The **count** *number* parameter specifies how many ping packets the router sends. You can specify from 1 - 4294967296. The default is 1.
- The **timeout** *milliseconds* parameter specifies how many milliseconds the router waits for a reply from the pinged device. You can specify a timeout from 1 - 4294967296 milliseconds. The default is 5000 (5 seconds).
- The **ttl** *number* parameter specifies the maximum number of hops. You can specify a TTL from 1 - 255. The default is 64.
- The **size** *bytes* parameter specifies the size of the ICMP data portion of the packet. This is the payload and does not include the header. You can specify from 0 - 10000. The default is 16.
- The **no-fragment** keyword turns on the "do not fragment" bit in the IPv6 header of the ping packet. This option is disabled by default.
- The **quiet** keyword hides informational messages such as a summary of the ping parameters sent to the device, and instead only displays messages indicating the success or failure of the ping. This option is disabled by default.
- The **verify** keyword verifies that the data in the echo packet (the reply packet) is the same as the data in the echo request (the ping). By default the device does not verify the data.
- The **data** *1 - 4 byte hex* parameter lets you specify a specific data pattern for the payload instead of the default data pattern, "abcd", in the packet's data payload. The pattern repeats itself throughout the ICMP message (payload) portion of the packet.

NOTE

For parameters that require a numeric value, the CLI does not check that the value you enter is within the allowed range. Instead, if you do exceed the range for a numeric value, the software rounds the value to the nearest valid value.

- The **brief** keyword causes ping test characters to be displayed. The following ping test characters are supported.

! Indicates that a reply was received.

. Indicates that the network server timed out while waiting for a reply.

U Indicates that a destination unreachable error PDU was received.

I Indicates that the user interrupted ping.

Configuring an IPv6 Syslog server

To enable IPv6 logging, specify an IPv6 Syslog server. Enter a command such as the following.

```
device(config)#log host ipv6 2000:2383:e0bb::4/128
```

Syntax: `log host ipv6 ipv6-address [udp-port-num]`

The `ipv6-address` must be in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The `udp-port-num` optional parameter specifies the UDP application port used for the Syslog facility.

Viewing IPv6 SNMP server addresses

Some of the **show** commands display IPv6 addresses for IPv6 SNMP servers. The following shows an example output for the **show snmp server** command.

```
device#show snmp server

Contact:
Location:
Community(ro): .....

Traps
    Warm/Cold start: Enable
    Link up: Enable
    Link down: Enable
    Authentication: Enable
    Locked address violation: Enable
    Power supply failure: Enable
    Fan failure: Enable
    Temperature warning: Enable
    STP new root: Enable
```



```

      STP topology change: Enable
      vsrp: Enable

Total Trap-Receiver Entries: 4

Trap-Receiver IP-Address      Port-Number Community
1      10.147.201.100
      162      .....
2      2001:DB8::200
      162      .....
3      10.147.202.100
      162      .....
4      2001:DB8::200
      162      .....

```

Disabling router advertisement and solicitation messages

Router advertisement and solicitation messages enable a node on a link to discover the routers on the same link. By default, router advertisement and solicitation messages are permitted on the device. To disable these messages, configure an IPv6 access control list that denies them. The following shows an example configuration.

```

device(config)#ipv6 access-list rtradvert
device(config)#deny icmp any any router-advertisement
device(config)#deny icmp any any router-solicitation
device(config)#permit ipv6 any any

```

Disabling IPv6 on a Layer 2 switch

IPv6 is enabled by default in the Layer 2 switch code. If desired, you can disable IPv6 on a global basis on a device running the switch code. To do so, enter the following command at the Global CONFIG level of the CLI.

```
device(config)#no ipv6 enable
```

Syntax: no ipv6 enable

To re-enable IPv6 after it has been disabled, enter **ipv6 enable** .

NOTE

IPv6 is disabled by default in the router code and must be configured on each interface that will support IPv6.

IPv6 ICMP feature configuration

As with the Internet Control Message Protocol (ICMP) for IPv4, ICMP for IPv6 provides error and informational messages. Implementation of the stateless auto configuration, neighbor discovery, and path MTU discovery features use ICMP messages.

This section explains how to configure following IPv6 ICMP features:

- ICMP rate limiting
- ICMP redirects

Configuring ICMP rate limiting

You can limit the rate at which IPv6 ICMP error messages are sent out on a network. IPv6 ICMP implements a token bucket algorithm.

To illustrate how this algorithm works, imagine a virtual bucket that contains a number of tokens. Each token represents the ability to send one ICMP error message. Tokens are placed in the bucket at a specified interval until the maximum number of tokens allowed in the bucket is reached. For each error message that ICMP sends, a token is removed from the bucket. If ICMP generates a series of error messages, messages can be sent until the bucket is empty. If the bucket is empty of tokens, error messages cannot be sent until a new token is placed in the bucket.

You can adjust the following elements related to the token bucket algorithm:

- The interval at which tokens are added to the bucket. The default is 100 milliseconds.
- The maximum number of tokens in the bucket. The default is 10 tokens.

For example, to adjust the interval to 1000 milliseconds and the number of tokens to 100 tokens, enter the following command.

```
device(config)# ipv6 icmp error-interval 1000 100
```

Syntax: `ipv6 icmp error-interval interval [number-of-tokens]`

The interval in milliseconds at which tokens are placed in the bucket can range from 0 - 2147483647. The maximum number of tokens stored in the bucket can range from 1 - 200.

NOTE

If you retain the default interval value or explicitly set the value to 100 milliseconds, output from the **show run** command does not include the setting of the **ipv6 icmp error-interval** command because the setting is the default. Also, if you configure the interval value to a number that does not evenly divide into 100000 (100 milliseconds), the system rounds up the value to a next higher value that does divide evenly into 100000. For example, if you specify an interval value of 150, the system rounds up the value to 200.

ICMP rate limiting is enabled by default. To disable ICMP rate limiting, set the interval to zero.

Enabling IPv6 ICMP redirect messages

You can enable a Layer 3 switch to send an IPv6 ICMP redirect message to a neighboring host to inform it of a better first-hop router on a path to a destination. By default, the sending of IPv6 ICMP redirect messages by a Layer 3 switch is disabled. (For more information about how ICMP redirect messages are implemented for IPv6, refer to [IPv6 neighbor discovery configuration](#) on page 147.)

NOTE

This feature is supported on Virtual Ethernet (VE) interfaces only.

For example, to enable the sending of IPv6 ICMP redirect messages on VE 2, enter the following commands.

```
device(config)#interface ve2
device(config-vif-2)#ipv6 redirects
```

To disable the sending of IPv6 ICMP redirect messages after it has been enabled on VE 2, enter the following commands.

```
device(config)#interface ve2
device(config-vif-2)#no ipv6 redirects
```

Syntax: `[no] ipv6 redirects`

Use the **show ipv6 interface** command to verify that the sending of IPv6 ICMP redirect messages is enabled on a particular interface.

IPv6 neighbor discovery configuration

The neighbor discovery feature for IPv6 uses IPv6 ICMP messages to do 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).

IPv6 neighbor discovery configuration notes

NOTE

For all solicitation and advertisement messages, Brocade uses seconds as the unit of measure instead of milliseconds.

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

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

- Neighbor discovery is not supported on tunnel interfaces.

Neighbor solicitation and advertisement messages

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

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

- Link-layer address of node 1.
- A query for the link-layer address of node 2.

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

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

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

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

Router advertisement and solicitation messages

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

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

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

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

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

Entering the **ipv6 unicast-routing** command automatically enables the sending of router advertisement messages on all configured router Ethernet interfaces. You can configure several router advertisement message parameters. For information about disabling the sending of router advertisement messages and the router advertisement parameters that you can configure, refer to [Enabling and disabling IPv6 router advertisements](#) on page 152 and [Setting IPv6 router advertisement parameters](#) on page 149.

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.

Setting neighbor solicitation parameters for duplicate address detection

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.

NOTE

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

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

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

For example, to change the number of neighbor solicitation messages sent on Ethernet interface 1/3/1 to two and the interval between the transmission of the two messages to 9 seconds, enter the following commands.

```
device(config)#interface ethernet 1/3/1
device(config-if-e1000-1/3/1)#ipv6 nd dad attempt 2
device(config-if-e1000-1/3/1)#ipv6 nd ns-interval 9000
```

Syntax: `[no] ipv6 nd dad attempt number`

Syntax: `[no] ipv6 nd ns-interval number`

For the number of neighbor solicitation messages, specify a number from 0 - 255. The default is 3. Configuring a value of 0 disables duplicate address detection processing on the specified interface. To restore the number of messages to the default value, use the **no** form of this command.

For the interval between neighbor solicitation messages and the value for the retrans timer in router advertisements, specify a number from 0 - 4294967295 milliseconds. The default value for the interval between neighbor solicitation messages is 1000 milliseconds. The default value for the retrans timer is 0. Brocade does not recommend very short intervals in normal IPv6 operation. When a non-default value is configured, the configured time is both advertised and used by the router itself. To restore the default interval, use the **no** form of this command.

Setting IPv6 router advertisement parameters

You can adjust the following parameters for router advertisement messages:

- The interval (in seconds) at which an interface sends router advertisement messages. By default, an interface sends a router advertisement message every 200 seconds.
- The "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.

When adjusting these parameter settings, Brocade 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 router. For example, to adjust the interval of router

advertisements to 300 seconds and the router lifetime value to 1900 seconds on Ethernet interface 1/3/1, enter the following commands.

```
device(config)#interface ethernet 1/3/1
device(config-if-e1000-1/3/1)#ipv6 nd ra-interval 300
device(config-if-e1000-1/3/1)#ipv6 nd ra-lifetime 1900
device(config-if-e1000-1/3/1)#ipv6 nd ra-hop-limit 1
```

Here is another example with a specified range.

```
device(config)#interface ethernet 1/3/1
device(config-if-e1000-1/3/1)#ipv6 nd ra-interval range 33 55
device(config-if-e1000-1/3/1)#ipv6 nd ra-lifetime 1900
device(config-if-e1000-1/3/1)#ipv6 nd ra-hop-limit 1
```

Syntax: [no] **ipv6 nd ra-interval** *number* | *min-range-value max-range-value*

Syntax: [no] **ipv6 nd ra-lifetime** *number*

Syntax: **ipv6 nd ra-hop-limit** *number*

number is a value from 0 - 255. The default is 64.

The **ipv6 nd ra-interval** *number* can be a value between 3 - 1800 seconds. The default is 200 seconds. The actual RA interval will be from .5 to 1.5 times the configured or default value. For example, in the above configuration, for **ipv6 nd ra-interval 300**, the range would be 150 - 450. To restore the default interval of 200 seconds, use the **no** form of the command.

The **ipv6 nd ra-interval range** *min range value max range value* command lets you specify a range of values instead of a single value.

The *min-range-value* specifies the minimum number of seconds allowed between sending unsolicited multicast router advertisements from the interface. The default is 0.33 times the *max-range-value* if the *max-range-value* is greater than or equal to 9 seconds. Otherwise, the default is the value specified by the *max-range-value*. The *min-range-value* can be a number between -3 - (.75 x *max range value*).

The *max-range-value* parameter specifies the maximum number of seconds allowed between sending unsolicited multicast router advertisements from the interface. This number can be between 4 - 1800 seconds and must be greater than the *min-range-value* x 1.33. The default is 600 seconds.

The **ipv6 nd ra-lifetime** *number* is a value between 0 - 9000 seconds. To restore the router lifetime value of 1800 seconds, use the **no** form of the command.

The **ipv6 nd ra-hop-limit** *number* is a value from 0 - 255. The default is 64.

NOTE

By default, router advertisements will always have the MTU option. To suppress the MTU option, use the following command at the Interface level of the CLI: **ipv6 nd suppress-mtu-option**.

Prefixes advertised in IPv6 router advertisement messages

By default, router advertisement messages include prefixes configured as addresses on router interfaces using the **ipv6 address** command. You can use the **ipv6 nd prefix-advertisement** command to control exactly which prefixes are included in router advertisement messages. Along with which prefixes the router advertisement messages contain, you can also specify the following parameters:

- **Valid lifetime** --(Mandatory) The time interval (in seconds) in which the specified prefix is advertised as valid. The default is 2592000 seconds (30 days). When the timer expires, the prefix is no longer considered to be valid.
- **Preferred lifetime** --(Mandatory) The time interval (in seconds) in which the specified prefix is advertised as preferred. The default is 604800 seconds (7 days). When the timer expires, the prefix is no longer considered to be preferred.

- **Onlink flag** --(Optional) If this flag is set, the specified prefix is assigned to the link upon which it is advertised. Nodes sending traffic to addresses that contain the specified prefix consider the destination to be reachable on the local link.
- **Autoconfiguration flag** --(Optional) If this flag is set, the stateless auto configuration feature can use the specified prefix in the automatic configuration of 128-bit IPv6 addresses for hosts on the local link, provided the specified prefix is aggregatable, as specified in RFC 2374.

For example, to advertise the prefix 2001:DB8:a487:7365::/64 in router advertisement messages sent out on Ethernet interface 1/3/1 with a valid lifetime of 1000 seconds, a preferred lifetime of 800 seconds, and the Onlink and Autoconfig flags set, enter the following commands.

```
device(config)#interface ethernet 1/3/1
device(config-if-e1000-1/3/1)#ipv6 nd prefix-advertisement 2001:DB8:a487:7365::/64 1000 800 onlink
autoconfig
```

Syntax: [no] **ipv6 nd prefix-advertisement** *ipv6-prefix/prefix-length valid-lifetime preferred-lifetime* [**autoconfig**] [**onlink**]

You must specify the *ipv6-prefix* parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

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.

The valid lifetime and preferred lifetime is a numerical value between 0 - 4294967295 seconds. The default valid lifetime is 2592000 seconds (30 days), while the default preferred lifetime is 604800 seconds (7 days).

To remove a prefix from the router advertisement messages sent from a particular interface, use the **no** form of this command.

Setting flags in IPv6 router advertisement messages

An IPv6 router advertisement message can include the following flags:

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

NOTE

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

By default, the Managed Address Configuration and Other Stateful Configuration flags are not set in router advertisement messages. For example, to set these flags in router advertisement messages sent from Ethernet interface 1/3/1, enter the following commands.

```
device(config)#interface ethernet 1/3/1
device(config-if-e1000-1/3/1)#ipv6 nd managed-config-flag
device(config-if-e1000-1/3/1)#ipv6 nd other-config-flag
```

Syntax: [no] **ipv6 nd managed-config-flag**

Syntax: [no] **ipv6 nd other-config-flag**

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

Enabling and disabling IPv6 router advertisements

If IPv6 unicast routing is enabled on an Ethernet interface, by default, this interface sends IPv6 router advertisement messages. However, by default, non-LAN interface types, for example, tunnel interfaces, do not send router advertisement messages.

To disable the sending of router advertisement messages on an Ethernet interface, enter commands such as the following.

```
device(config)#interface ethernet 1/3/1
device(config-if-e1000-1/3/1)#ipv6 nd suppress-ra
```

To enable the sending of router advertisement messages on a tunnel interface, enter commands such as the following.

```
device(config)#interface tunnel 1
device(config-tnif-1)#no ipv6 nd suppress-ra
```

Syntax: [no] ipv6 nd suppress-ra

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.

To configure IPv6 RA preference for the IPv6 router, use the **ipv6 nd router-preference** in the interface configuration mode.

The following example shows the router preference configured for interface 1/2/3 with the preference value "low".

```
device(config)#interface ethernet 1/2/3
device(config-if-e1000-1/2/3)#ipv6 nd router-preference low
```

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 the value of 30 seconds.

The router advertisement messages sent by a router interface include the amount of time specified by the **ipv6 nd reachable-time** command so that nodes on a link use the same reachable time duration. By default, the messages include a default value of 0.

Brocade does not recommend configuring a short reachable time duration, because a short duration causes the IPv6 network devices to process the information at a greater frequency.

For example, to configure the reachable time of 40 seconds for Ethernet interface 1/3/1, enter the following commands.

```
device(config)#interface ethernet 1/3/1
device(config-if-e1000-1/3/1)#ipv6 nd reachable-time 40
```

Syntax: [no] ipv6 nd reachable-time *seconds*

For the *seconds* variable, specify a number from 0 through 3600 seconds. To restore the default time, use the **no** form of this command.

NOTE

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

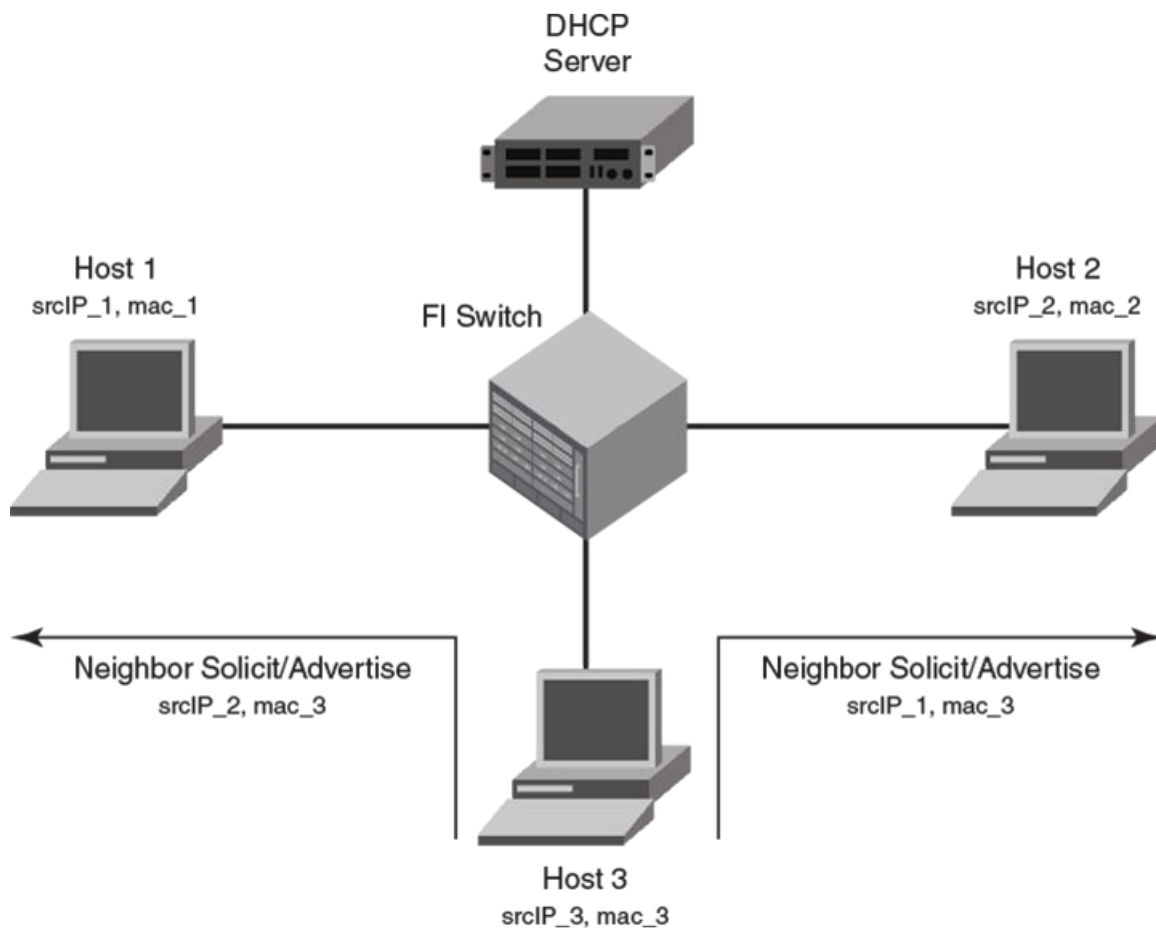
IPv6 neighbor discovery inspection

IPv6 ND inspection is an internal network security system that detects and prevents IPv6 address spoofing at the switch level.

IP communication within a Layer 2 infrastructure is established by mapping an IP address to a MAC address. An invalid host can intercept packet flow between legitimate hosts by sending a neighbor solicitation or neighbor advertisement with a forged IP-to-MAC address binding. The victim host includes an illegitimate entry in the neighbor cache, which is looked up to validate the IP-to-MAC address binding. After a successful attack, all the traffic will be redirected through the invalid host and is vulnerable to man-in-the-middle attacks. The ND inspection validates all the IPv6 packets carrying neighbor discovery messages by checking the IP-to-MAC address binding of the packets. If there is a discrepancy in the IP-to-MAC address binding, the neighbor discovery message is considered to be from an invalid host and the packets are discarded.

The following figure illustrates the method by which Host 3 performs ND cache poisoning by sending a neighbor solicitation message to Host 1 with the source IP of Host 2, and similarly to Host 2 with the source IP of Host 1, with its own MAC address. By doing this, Host 3 can intercept the packet flow from Host 1 to Host 2.

FIGURE 12 Neighbor discovery cache poisoning



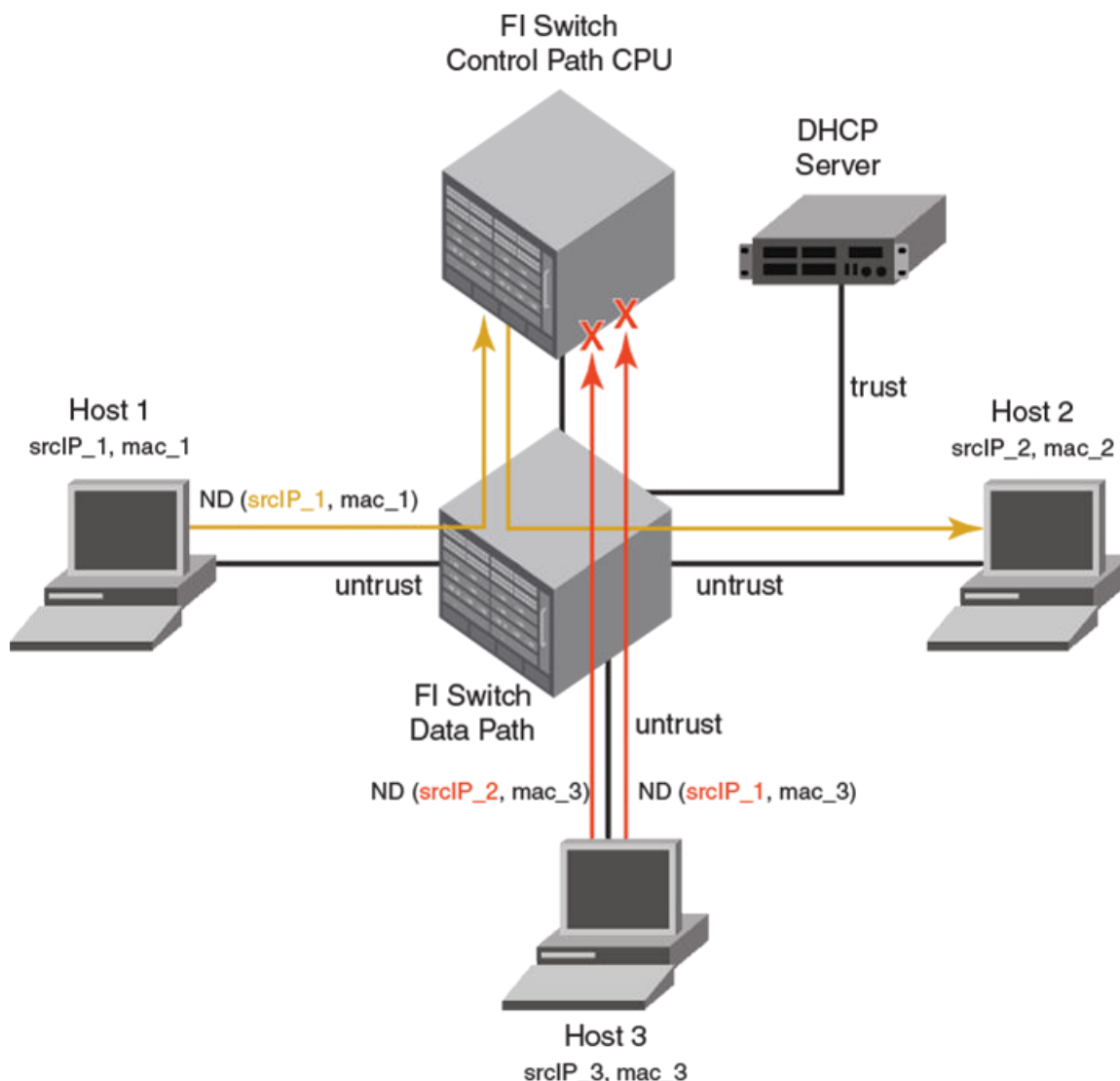
ND inspection, when enabled on a VLAN, checks all the neighbor discovery messages flowing through the switches between the hosts that are part of the VLAN and validates the IP-to-MAC address binding of the packets. All the packets are verified against the trusted binding tables where the preconfigured static ND inspection entries or dynamically learned DHCPv6 snoop entries are stored. DHCPv6 snooping must be enabled for dynamic inspection of ND messages. For more information on dynamically learned DHCPv6 snoop entries, refer to the DHCPv6 section in the *Brocade FastIron Configuration Guide*.

To inspect a neighbor discovery message, all the neighbor solicitation and neighbor advertisement messages are directed to a CPU, and the source IP address and source MAC address of each packet are validated against the entries in the trusted tables. Only the valid packets are forwarded and those with invalid IP-to-MAC address bindings are discarded. ND inspection follows CPU-based packet forwarding and thus the neighbor discovery messages in the ND inspection-enabled VLAN may get discarded depending on the CPU load. The neighbor discovery messages are also rate limited to CPU.

The router interface configuration on the ND inspection-enabled VLAN is also subjected to ND inspection. That is, if the interface is a Layer 3 interface, the neighbor solicitation and neighbor advertisement messages addressed to the router are also validated. If there is a discrepancy in the IP-to-MAC address binding, the packets are discarded and the IPv6 neighbor tables will not be updated. Unlike the neighbor solicitation and neighbor advertisement messages, the router solicitation messages are not directed to the CPU, because the hosts are supposed to reject the router solicitation messages by default.

The following figure illustrates unhindered flow of packets from Host 1 to Host 2, while the messages that are sent by Host 3 with invalid IP-to-MAC address bindings are discarded.

FIGURE 13 Neighbor discovery inspection



Though you can configure interfaces in “trust” or “untrust” mode, ND inspection is performed only on untrusted ports that are part of the ND inspection-enabled VLAN. When you enable ND inspection on a VLAN, by default, all the interfaces and member ports are considered as untrusted. When configured, ND inspection protects the directly connected hosts from ND cache poisoning; the hosts connected across the switches are not insulated from any attack.

When configured, ND inspection performs the following functions:

- Intercepts and inspects the IPv6 packets that carry neighbor discovery messages on untrusted ports.
- Validates the source IP addresses and the source MAC addresses of the intercepted packets against the IP-to-MAC address bindings stored in a trusted binding database.
- Forwards the packets which have valid IP-to-MAC address bindings to the destination host and discards the invalid packets. The ICMPv6 packets with auto-generated link-local address (from the MAC address) are also forwarded, provided there is a match between MAC address and the auto-generated link-local address. Hence, there is no need of separate configuration of auto-generated link-local address in the ND inspection database.

NOTE

ND inspection is supported on LAGs and trunk ports and supports Multi-VRF instances. Multiple VRFs can be deployed on a Brocade Ethernet switch. Each VLAN having a Virtual Interface (VE) is assigned to a VRF.

Neighbor discovery inspection configuration

The ND inspection configuration includes enabling ND inspection on a VLAN, adding static inspection entries, and enabling trust mode for switch or server ports.

The `acl-per-port-per-vlan` must be enabled (using **enable acl-per-port-per-vlan**) command before configuring ND inspection.

1. Enter the **ipv6 neighbor inspection vlan** *vlan-number* command to enable ND inspection on a VLAN.
2. Enter the **ipv6 neighbor inspection** *ipv6-address mac-address* command to add a static ND inspection entry. You can add multiple static ND inspection entries.
3. Enter the **interface ethernet** command to enter the interface configuration mode.
4. Enter the **ipv6-neighbor inspection trust** command to enable trust mode for the switch or server port. You can enable trust mode for multiple ports.

The following output shows an example of ND inspection configuration.

```
device(config)# ipv6 neighbor inspection vlan 10
device(config)# ipv6 neighbor inspection 2001::1 0000.1234.5678
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# ipv6-neighbor inspection trust
```

Syslog message for ND inspection

The following table lists the syslog message related to ND inspection.

TABLE 26 Syslog message related to ND inspection

Event	Syslog output
Rejected ND	ND Inspect: no static inspect or DHCP6 entry found, packet dropped rx-sip 2001::100 rx-smac 0000.0000.0055 vlan_id 2 vrf_id 0

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.

By default, in non-jumbo mode, the default and maximum Ethernet MTU size is 1500 bytes. When jumbo mode is enabled, the default Ethernet MTU size is 9216. The maximum Ethernet MTU size is 10128.

Configuration notes and feature limitations for IPv6 MTU

- The IPv6 MTU functionality is applicable to VEs and physical IP interfaces. It applies to traffic routed between networks.
- For ICX 7250, ICX 7450, and ICX 7750 devices, the IPv4 and IPv6 MTU values are the same. Modifying one also changes the value of the other.
- For ICX 7250, ICX 7450, and ICX 7750 devices, the minimum IPv4 and IPv6 MTU values for both physical and virtual interfaces are 1280.

- You cannot use IPv6 MTU to set Layer 2 maximum frame sizes per interface. Enabling global jumbo mode causes all interfaces to accept Layer 2 frames.

Changing the IPv6 MTU

You can configure the IPv6 MTU on individual interfaces. For example, to configure the MTU on Ethernet interface 1/3/1 as 1280 bytes, enter the following commands.

```
device(config)# interface ethernet 1/3/1
device(config-if-e1000-1/3/1)# ipv6 mtu 1280
```

Syntax: **[no]** **ipv6 mtu** *bytes*

For bytes, specify a value between 1280 - 1500, or 1280 - 10218 if jumbo mode is enabled. If a non-default value is configured for an interface, router advertisements include an MTU option.

NOTE

IPv6 MTU cannot be configured globally. It is supported only on devices running Layer 3 software.

Static neighbor entries configuration

In some special cases, a neighbor cannot be reached using the neighbor discovery feature. In this situation, you can add a static entry to the IPv6 neighbor discovery cache, which causes a neighbor to be reachable at all times without using neighbor discovery. (A static entry in the IPv6 neighbor discovery cache functions like a static ARP entry in IPv4.)

NOTE

A port that has a statically assigned IPv6 entry cannot be added to a VLAN.

NOTE

Static neighbor configurations will be cleared on secondary ports when a LAG is formed.

For example, to add a static entry for a neighbor with the IPv6 address 2001:DB8:2678:47b and link-layer address 0000.002b.8641 that is reachable through Ethernet interface 1/3/1, enter the `ipv6 neighbor` command.

```
device(config)# ipv6 neighbor 2001:DB8:2678:47b ethernet 1/3/1 0000.002b.8641
```

Syntax: **[no]** **ipv6 neighbor** *ipv6-address* **ethernet** *port* | **ve** *ve-number* [**ethernet** *port*] *link-layer-address*

The *ipv6-address* parameter specifies the address of the neighbor.

The **ethernet** | **ve** parameter specifies the interface through which to reach a neighbor. If you specify an Ethernet interface, specify the port number of the Ethernet interface. If you specify a VE, specify the VE number and then the Ethernet port numbers associated with the VE. The link-layer address is a 48-bit hardware address of the neighbor.

If you attempt to add an entry that already exists in the neighbor discovery cache, the software changes the already existing entry to a static entry.

To remove a static IPv6 entry from the IPv6 neighbor discovery cache, use the **no** form of this command.

Limiting the number of hops an IPv6 packet can traverse

Limiting the number of hops an IPv6 packet can traverse

By default, the maximum number of hops an IPv6 packet can traverse is 64. You can change this value to between 0 - 255 hops. For example, to change the maximum number of hops to 70, enter the following command.

```
device(config)#ipv6 hop-limit 70
```

Syntax: **[no] ipv6 hop-limit** *number*

Use the **no** form of the command to restore the default value.

hop-limit 0 will transmit packets with default (64) hop limit.

number can be from 0 - 255.

IPv6 source routing security enhancements

The IPv6 specification (RFC 2460) specifies support for IPv6 source-routed packets using a type 0 Routing extension header, requiring device and host to process the type 0 routing extension header. However, this requirement may leave a network open to a DoS attack.

A security enhancement disables sending IPv6 source-routed packets to IPv6 devices. (This enhancement conforms to RFC 5095.)

By default, when the router drops a source-routed packet, it sends an ICMP Parameter Problem (type 4), Header Error (code 0) message to the packet's source address, pointing to the unrecognized routing type. To disable these ICMP error messages, enter the following command:

```
device(config)# no ipv6 icmp source-route
```

Syntax: **[no] ipv6 icmp source-route**

Use the **ipv6 icmp source-route** form of the command to enable the ICMP error messages.

TCAM space configuration

Brocade devices store routing information for IPv4 and IPv6 routing and GRE tunnel information in the same ternary content-addressable memory (TCAM) table. You can configure the amount of TCAM space to allocate for IPv4 routing information and GRE tunnels. The remaining space is allocated automatically for IPv6 routing information.

Brocade devices vary in the amount of TCAM space that can be allocated for IPv4 and IPv6 routing and GRE tunnel information. Each IPv6 route entry and GRE tunnel use more storage space than IPv4 route entries. The default, maximum, and minimum allocation values for each type of data are shown in the following tables.

NOTE

If you disable IPv6 routing, the TCAM space allocations do not change. If you want to allocate the maximum possible space for IPv4 routing information, you must configure the TCAM space manually.

TABLE 27 TCAM space allocation on ICX 7750 devices

	Default	Maximum	Minimum
IPv4 route entries	12000	15168	4096
IPv6 route entries	5120	5120	68

TABLE 27 TCAM space allocation on ICX 7750 devices (continued)

	Default	Maximum	Minimum
GRE tunnels	16	64	16

TABLE 28 TCAM space allocation on ICX 7450 devices

	Default	Maximum	Minimum
IPv4 route entries	12000	15168	4096
IPv6 route entries	5120	5120	68
GRE tunnels	16	64	16

TABLE 29 TCAM space allocation on ICX 7250 devices

	Default	Maximum	Minimum
IPv4 route entries	6000	6000	2048
IPv6 route entries	365	1024	365
GRE tunnels	8	8	8

NOTE

The ICX 7250 device has a fixed allocation of space for eight GRE tunnels.

Allocating TCAM space

The amount of TCAM space to allocate for IPv4 routing information can be configured. You must save the running configuration to the startup configuration and reload the device for the changes to take effect. After the reload, the remaining TCAM space is allocated automatically for IPv6 routing information.

TCAM space allocations for IPv4 and IPv6 routes and other entities can be modified by configuring the number of IPv4 route entries. Different devices have different amounts of TCAM space, see the "TCAM space allocation" topic for the default, maximum, and minimum allocations.

NOTE

If you disable IPv6 routing, the TCAM space allocations do not change. If you want to allocate the maximum possible space for IPv4 routing information, you must configure the TCAM space manually.

NOTE

The ICX 7250 device only permits manual configuration of IPv4 routes.

1. Enter global configuration mode.

```
device# configure terminal
```

2. To allocate TCAM space to store 6000 IPv4 routes entries, use the following command.

```
device(config)# system-max ip-route 6000
```

3. Copy the running configuration to the startup configuration.

```
device(config)# write memory
```

4. Return to privileged EXEC mode.

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

5. Reload the device for the new TCAM space allocations to be changed.

```
device# reload
```

The following example configures TCAM space for 6000 IPv4 route entries. After the reload, you can view the new TCAM allocation numbers for IPv6 entries.

```
device# configure terminal
device(config)# system-max ip-route 6000
device(config)# write memory
device(config)# exit
device# reload
device# show run
.
.
.
ip-route: 6000
ip6-route 365
ip6-cache: 182
```

Allocating TCAM space for GRE tunnels

The amount of TCAM space to allocate for Generic Routing Encapsulation (GRE) tunnel information can be configured. You must save the running configuration to the startup configuration and reload the device for the changes to take effect. After the reload, the remaining TCAM space is allocated automatically for IPv6 routing information.

TCAM space allocations for GRE tunnels can be modified using manual configuration. Different devices have different amounts of TCAM space, see the "TCAM space allocation" topic for the default, maximum, and minimum allocations. The TCAM space allocation is dependent on the IPv4 route configuration.

NOTE

This task is not supported on the ICX 7250 device where the TCAM allocation is for 8 GRE tunnels.

1. Enter global configuration mode.

```
device# configure terminal
```

2. To allocate TCAM space to store information for up to 20 GRE tunnels, use the following command.

```
device(config)# system-max gre-tunnels 20
```

3. Copy the running configuration to the startup configuration.

```
device(config)# write memory
```

4. Return to privileged EXEC mode.

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

5. Reload the device for the new TCAM space allocations to be changed.

```
device# reload
```


The following example configures TCAM storage space for 20 GRE tunnel entries. After the reload, you can view the new TCAM allocation numbers for GRE tunnels in the running configuration.

```
device# configure terminal
device(config)# system-max gre-tunnels 20
device(config)# write memory
device(config)# exit
device# reload
device# show run
.
.
.
gre-tunnels: 20
ip-route: 6000
ip6-route 365
ip6-cache: 182
```

Displaying global IPv6 information

You can display output for the following global IPv6 parameters:

- IPv6 cache
- IPv6 interfaces
- IPv6 neighbors
- IPv6 route table
- Local IPv6 routers
- IPv6 TCP connections and the status of individual connections
- IPv6 traffic statistics

Displaying IPv6 cache information

The IPv6 cache contains an IPv6 host table that has indices to the next hop gateway and the router interface on which the route was learned.

To display IPv6 cache information, enter the following command at any CLI level.

```
device# show ipv6 cache
Total number of cache entries: 10
  IPv6 Address      Next Hop      Port
1   2001:DB8::2      LOCAL         tunnel 2
2   2001:DB8::106    LOCAL         ethe 1/3/2
3   2001:DB8::110    DIRECT        ethe 1/3/2
4   2001:DB8:46a::1  LOCAL         ethe 1/3/2
5   2001:DB8::2e0:52ff:fe99:9737 LOCAL         ethe 1/3/2
6   2001:DB8::ffff:ffff:feff:ffff LOCAL         loopback 2
7   2001:DB8::c0a8:46a LOCAL         tunnel 2
8   2001:DB8::c0a8:46a LOCAL         tunnel 6
9   2001:DB8::1      LOCAL         loopback 2
10  2001:DB8::2e0:52ff:fe99:9700 LOCAL         ethe 1/3/1
```

Syntax: `show ipv6 cache` [*index-number* | *ipv6-prefix/prefix-length* | *ipv6-address* | **ethernet** *unit / slot / port* | **venumber** | **tunnel** *number*]

The *index-number* parameter restricts the display to the entry for the specified index number and subsequent entries.

The *ipv6-prefix/prefix-length* parameter restricts the display to the entries for the specified IPv6 prefix. You must specify the *ipv6-prefix* parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. 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.

Displaying global IPv6 information

The **ethernet | ve | tunnel** parameter restricts the display to the entries for the specified interface. The *ipv6-address* parameter restricts the display to the entries for the specified IPv6 address. You must specify this parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

If you specify an Ethernet interface, also specify the *unit / slot / port* number associated with the interface. If you specify a VE interface, also specify the VE number. If you specify a tunnel interface, also specify the tunnel number.

This display shows the following information.

TABLE 30 IPv6 cache information fields

Field	Description
Total number of cache entries	The number of entries in the cache table.
IPv6 Address	The host IPv6 address.
Next Hop	The next hop, which can be one of the following: <ul style="list-style-type: none"> • Direct - The next hop is directly connected to the router. • Local - The next hop is originated on this router. • ipv6 address - The IPv6 address of the next hop.
Port	The port on which the entry was learned.

Displaying IPv6 interface information

To display IPv6 interface information, enter the following command at any CLI level.

```
device#show ipv6 interface
Routing Protocols : R - RIP  O - OSPF
Interface          Status      Routing   Global Unicast Address
Ethernet 1/3/3      down/down  R
Ethernet 1/3/5      down/down
Ethernet 1/3/17     up/up      2017::c017:101/64
Ethernet 1/3/19     up/up      2019::c019:101/64
VE 4                down/down
VE 14               up/up      2024::c060:101/64
Loopback 1          up/up      ::1/128
Loopback 2          up/up      2005::303:303/128
Loopback 3          up/up
```

Syntax: `show ipv6 interface [interface [unit / slot / port | number]]`

The interface parameter displays detailed information for a specified interface. For the interface, you can specify the **Ethernet** , **loopback** , **tunnel** , or **VE** keywords. If you specify an Ethernet interface, also specify *unit / slot / port* . If you specify a loopback, tunnel, or VE interface, also specify the number associated with the interface.

This display shows the following information.

TABLE 31 General IPv6 interface information fields

Field	Description
Routing protocols	A one-letter code that represents a routing protocol that can be enabled on an interface.
Interface	The interface type, and the port number or number of the interface.
Status	The status of the interface. The entry in the Status field will be either "up/up" or "down/down".
Routing	The routing protocols enabled on the interface.
Global Unicast Address	The global unicast address of the interface.

To display detailed information for a specific interface, enter a command such as the following at any CLI level.

```
device#show ipv6 interface ethernet 1/3/1
Interface Ethernet 1/3/1 is up, line protocol is up
IPv6 is enabled, link-local address is fe80::2e0:52ff:fe99:97
Global unicast address(es):
Joined group address(es):
  ff02::9
  ff02::1:ff99:9700
  ff02::2
  ff02::1
MTU is 1500 bytes
ICMP redirects are enabled
ND DAD is enabled, number of DAD attempts: 3
ND reachable time is 30 seconds
ND advertised reachable time is 0 seconds
ND retransmit interval is 1 seconds
ND advertised retransmit interval is 0 seconds
ND router advertisements are sent every 200 seconds
ND router advertisements live for 1800 seconds
No Inbound Access List Set
No Outbound Access List Set
RIP enabled
```

This display shows the following information.

TABLE 32 Detailed IPv6 interface information fields

Field	Description
Interface/line protocol status	The status of interface and line protocol. If you have disabled the interface with the disable command, the status will be "administratively down". Otherwise, the status is either "up" or "down".
IPv6 status/link-local address	The status of IPv6. The status is either "enabled" or "disabled". Displays the link-local address, if one is configured for the interface.
Global unicast address(es)	Displays the global unicast address(es), if one or more are configured for the interface.
Joined group address(es)	The multicast address(es) that a router interface listens for and recognizes.
MTU	The setting of the maximum transmission unit (MTU) configured for the IPv6 interface. The MTU is the maximum length an IPv6 packet can have to be transmitted on the 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.
ICMP	The setting of the ICMP redirect parameter for the interface.
ND	The setting of the various neighbor discovery parameters for the interface.
Access List	The inbound and outbound access control lists applied to the interface.
Routing protocols	The routing protocols enabled on the interface.

Displaying IPv6 neighbor information

You can display the IPv6 neighbor table, which contains an entry for each IPv6 neighbor with which the router exchanges IPv6 packets.

To display the IPv6 neighbor table, enter the following command at any CLI level.

```
device(config)# show ipv6 neighbor
Total number of Neighbor entries: 3
IPv6 Address      LinkLayer-Addr  State      Age    Port      vlan  IsR
2001:DB8::55      0000.0002.0002  *REACH     0      e 1/3/11   -     0
2000:4::110       0000.0091.bb37  REACH      20     e 1/3/1    5     1
fe80::2e0:52ff:fe91:bb37 0000.0091.bb37  DELAY      1      e 1/3/2    4     1
fe80::2e0:52ff:fe91:bb40 0000.0091.bb40  STALE     5930   e 1/3/3    5     1
```

Syntax: `show ipv6 neighbor [ipv6-prefix/prefix-length | ipv6-address | interface [unit / slot / port | number]]`

The `ipv6-prefix / prefix-length` parameters restrict the display to the entries for the specified IPv6 prefix. You must specify the `ipv6-prefix` parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. 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.

The `ipv6-address` parameter restricts the display to the entries for the specified IPv6 address. You must specify this parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The `interface` parameter restricts the display to the entries for the specified router interface. For this parameter, you can specify the **ethernet** or **ve** keywords. If you specify an Ethernet interface, also specify `unit / slot / port`. If you specify a VE interface, also specify the VE number.

This display shows the following information.

TABLE 33 IPv6 neighbor information fields

Field	Description
Total number of neighbor entries	The total number of entries in the IPv6 neighbor table.
IPv6 Address	The 128-bit IPv6 address of the neighbor.
Link-Layer Address	The 48-bit interface ID of the neighbor.
State	The current state of the neighbor. Possible states are as follows: <ul style="list-style-type: none"> INCOMPLETE - Address resolution of the entry is being performed. *REACH - The static forward path to the neighbor is functioning properly. REACH - The forward path to the neighbor is functioning properly. STALE - This entry has remained unused for the maximum interval. While stale, no action takes place until a packet is sent. DELAY - This entry has remained unused for the maximum interval, and a packet was sent before another interval elapsed. PROBE - Neighbor solicitation are transmitted until a reachability confirmation is received.
Age	The number of seconds the entry has remained unused. If this value remains unused for the number of seconds specified by the ipv6 nd reachable-time command (the default is 30 seconds), the entry is removed from the table.
Port	The physical port on which the entry was learned.
vlan	The VLAN on which the entry was learned.
IsR	Determines if the neighbor is a router or host: <ul style="list-style-type: none"> 0 - Indicates that the neighbor is a host. 1 - Indicates that the neighbor is a router.

Displaying the IPv6 route table

To display the IPv6 route table, use the **show ipv6 route** command.

```
device# show ipv6 route
IPv6 Routing Table - 7 entries:
Type Codes: C - Connected, S - Static, R - RIP, O - OSPF, B - BGP
OSPF Sub Type Codes: O - Intra, Oi - Inter, O1 - Type1 external, O2 - Type2 external
Type  IPv6 Prefix      Next Hop Router      Interface      Dis/Metric
C      2000:4::/64         ::                  ethe 1/3/2     0/0
S      2001:DB8::/16       ::                  tunnel 6       1/1
```

```

S      2001:DB8:1234::/32      ::      tunnel 6      1/1
C      2001:DB8:46a::/64       ::      ethe 1/3/2    0/0
C      2001:DB8::1/128         ::      loopback 2    0/0
O      2001:DB8::2/128         fe80::2e0:52ff:fe91:bb37 ethe 1/3/2    110/1
C      2001:DB8::/64          ::      tunnel 2      0/0

```

The **ipv6-address** parameter restricts the display to the entries for the specified IPv6 address. You must specify the **ipv6-address** parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **ipv6-prefix / prefix-length** parameters restrict the display to the entries for the specified IPv6 prefix. You must specify the **ipv6-prefix** parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. 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.

The **bgp** keyword restricts the display to entries for BGP4 routes.

The **connect** keyword restricts the display to entries for directly connected interface IPv6 routes.

The **ospf** keyword restricts the display to entries for OSPFv3 routes.

The **rip** keyword restricts the display to entries for RIPng routes.

The **static** keyword restricts the display to entries for static IPv6 routes.

The **summary** keyword displays a summary of the prefixes and different route types.

The following table lists the information displayed by the **show ipv6 route** command.

TABLE 34 IPv6 route table fields

Field	Description
Number of entries	The number of entries in the IPv6 route table.
Type	The route type, which can be one of the following: <ul style="list-style-type: none"> C - The destination is directly connected to the router. S - The route is a static route. R - The route is learned from RIPng. O - The route is learned from OSPFv3. B - The route is learned from BGP4.
IPv6 Prefix	The destination network of the route.
Next-Hop Router	The next-hop router.
Interface	The interface through which this router sends packets to reach the route's destination.
Dis/Metric	The route's administrative distance and metric value.

To display a summary of the IPv6 route table, enter the **show ipv6 route summary** command.

```

device#show ipv6 route summary
IPv6 Routing Table - 7 entries:
 4 connected, 2 static, 0 RIP, 1 OSPF, 0 BGP
Number of prefixes:
/16: 1 /32: 1 /64: 3 /128: 2

```

The command-line interface (CLI) output of the **show ipv6 route summary** command displays the following information.

TABLE 35 IPv6 route table summary fields

Field	Description
Number of entries	The number of entries in the IPv6 route table.
Number of route types	The number of entries for each route type.
Number of prefixes	A summary of prefixes in the IPv6 route table, sorted by prefix length.

Displaying global IPv6 information

Displaying local IPv6 routers

The Brocade device can function as an IPv6 host, instead of an IPv6 router, if you configure IPv6 addresses on its interfaces but do not enable IPv6 routing using the **ipv6 unicast-routing** command.

From the IPv6 host, you can display information about IPv6 routers to which the host is connected. The host learns about the routers through their router advertisement messages. To display information about the IPv6 routers connected to an IPv6 host, enter the following command at any CLI level.

```
device#show ipv6 router
Router fe80::2e0:80ff:fe46:3431 on Ethernet 50, last update 0 min
Hops 64, Lifetime 1800 sec
Reachable time 0 msec, Retransmit time 0 msec
```

Syntax: show ipv6 router

If you configure your Brocade device to function as an IPv6 router (you configure IPv6 addresses on its interfaces and enable IPv6 routing using the **ipv6 unicast-routing** command) and you enter the **show ipv6 router command**, you will receive the following output.

```
No IPv6 router in table
```

Meaningful output for this command is generated for Brocade devices configured to function as IPv6 hosts only.

This display shows the following information.

TABLE 36 IPv6 local router information fields

Field	Description
Router ipv6 address on interface port	The IPv6 address for a particular router interface.
Last update	The amount of elapsed time (in minutes) between the current and previous updates received from a router.
Hops	The default value that should be included in the Hop Count field of the IPv6 header for outgoing IPv6 packets. The hops value applies to the router for which you are displaying information and should be followed by IPv6 hosts attached to the router. A value of 0 indicates that the router leaves this field unspecified.
Lifetime	The amount of time (in seconds) that the router is useful as the default router.
Reachable time	The amount of time (in milliseconds) that a router assumes a neighbor is reachable after receiving a reachability confirmation. The reachable time value applies to the router for which you are displaying information and should be followed by IPv6 hosts attached to the router. A value of 0 indicates that the router leaves this field unspecified.
Retransmit time	The amount of time (in milliseconds) between retransmissions of neighbor solicitation messages. The retransmit time value applies to the router for which you are displaying information and should be followed by IPv6 hosts attached to the router. A value of 0 indicates that the router leaves this field unspecified.

Displaying IPv6 TCP information

You can display the following IPv6 TCP information:

- General information about each TCP connection on the router, including the percentage of free memory for each of the internal TCP buffers.
- Detailed information about a specified TCP connection.

To display general information about each TCP connection on the router, enter the following command at any CLI level.

```
device#show ipv6 tcp connections
Local IP address:port <-> Remote IP address:port TCP state
10.168.182.110:23 <-> 10.168.8.186:4933 ESTABLISHED
10.168.182.110:8218 <-> 10.168.182.106:179 ESTABLISHED
10.168.182.110:8039 <-> 10.168.2.119:179 SYN-SENT
10.168.182.110:8159 <-> 10.168.2.102:179 SYN-SENT
2000:4::110:179 <-> 2000:4::106:8222 ESTABLISHED (1440)
Total 5 TCP connections
TCP MEMORY USAGE PERCENTAGE
FREE TCP = 98 percent
FREE TCP QUEUE BUFFER = 99 percent
FREE TCP SEND BUFFER = 97 percent
FREE TCP RECEIVE BUFFER = 100 percent
FREE TCP OUT OF SEQUENCE BUFFER = 100 percent
```

Syntax: show ipv6 tcp connections

This display shows the following information.

TABLE 37 General IPv6 TCP connection fields

Field	Description
Local IP address:port	The IPv4 or IPv6 address and port number of the local router interface over which the TCP connection occurs.
Remote IP address:port	The IPv4 or IPv6 address and port number of the remote router interface over which the TCP connection occurs.
TCP state	The state of the TCP connection. Possible states include the following: <ul style="list-style-type: none"> • LISTEN - Waiting for a connection request. • SYN-SENT - Waiting for a matching connection request after having sent a connection request. • SYN-RECEIVED - Waiting for a confirming connection request acknowledgment after having both received and sent a connection request. • ESTABLISHED - Data can be sent and received over the connection. This is the normal operational state of the connection. • FIN-WAIT-1 - Waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request previously sent. • FIN-WAIT-2 - Waiting for a connection termination request from the remote TCP. • CLOSE-WAIT - Waiting for a connection termination request from the local user. • CLOSING - Waiting for a connection termination request acknowledgment from the remote TCP. • LAST-ACK - Waiting for an acknowledgment of the connection termination request previously sent to the remote TCP (which includes an acknowledgment of its connection termination request). • TIME-WAIT - Waiting for enough time to pass to be sure the remote TCP received the acknowledgment of its connection termination request. • CLOSED - There is no connection state.
FREE TCP = percentage	The percentage of free TCP control block (TCP) space.
FREE TCP QUEUE BUFFER = percentage	The percentage of free TCP queue buffer space.
FREE TCP SEND BUFFER = percentage	The percentage of free TCP send buffer space.

Displaying global IPv6 information

TABLE 37 General IPv6 TCP connection fields (continued)

Field	Description
FREE TCP RECEIVE BUFFER = percentage	The percentage of free TCP receive buffer space.
FREE TCP OUT OF SEQUENCE BUFFER = percentage	The percentage of free TCP out of sequence buffer space.

To display detailed information about a specified TCP connection, enter a command such as the following at any CLI level.

```
device#show ipv6 tcp status 2000:4::110 179 2000:4::106 8222
TCP: TCP = 0x217fc300
TCP: 2000:4::110:179 <-> 2000:4::106:8222: state: ESTABLISHED Port: 1
  Send: initial sequence number = 242365900
  Send: first unacknowledged sequence number = 242434080
  Send: current send pointer = 242434080
  Send: next sequence number to send = 242434080
  Send: remote received window = 16384
  Send: total unacknowledged sequence number = 0
  Send: total used buffers 0
  Receive: initial incoming sequence number = 740437769
  Receive: expected incoming sequence number = 740507227
  Receive: received window = 16384
  Receive: bytes in receive queue = 0
  Receive: congestion window = 1459
```

Syntax: `show ipv6 tcp status local-ip-address local-port-number remote-ip-address remote-port-number`

The local-ip-address parameter can be the IPv4 or IPv6 address of the local interface over which the TCP connection is taking place.

The local-port-number parameter is the local port number over which a TCP connection is taking place.

The remote-ip-address parameter can be the IPv4 or IPv6 address of the remote interface over which the TCP connection is taking place.

The remote-port-number parameter is the local port number over which a TCP connection is taking place.

This display shows the following information.

TABLE 38 Specific IPv6 TCP connection fields

Field	Description
TCP = location	The location of the TCP.
local-ip-address local-port-number remote-ip-address remote-port-number state port	This field provides a general summary of the following: <ul style="list-style-type: none"> The local IPv4 or IPv6 address and port number. The remote IPv4 or IPv6 address and port number. The state of the TCP connection. For information on possible states, refer to Displaying IPv6 TCP information. The port numbers of the local interface.
Send: initial sequence number = number	The initial sequence number sent by the local router.
Send: first unacknowledged sequence number = number	The first unacknowledged sequence number sent by the local router.
Send: current send pointer = number	The current send pointer.
Send: next sequence number to send = number	The next sequence number sent by the local router.
Send: remote received window = number	The size of the remote received window.
Send: total unacknowledged sequence number = number	The total number of unacknowledged sequence numbers sent by the local router.
Send: total used buffers number	The total number of buffers used by the local router in setting up the TCP connection.
Receive: initial incoming sequence number = number	The initial incoming sequence number received by the local router.
Receive: expected incoming sequence number = number	The incoming sequence number expected by the local router.

TABLE 38 Specific IPv6 TCP connection fields (continued)

Field	Description
Receive: received window = number	The size of the local router's receive window.
Receive: bytes in receive queue = number	The number of bytes in the local router's receive queue.
Receive: congestion window = number	The size of the local router's receive congestion window.

Displaying IPv6 traffic statistics

To display IPv6 traffic statistics, enter the following command at any CLI level.

```
device#show ipv6 traffic
IPv6 Statistics
 36947 received, 66818 sent, 0 forwarded, 36867 delivered, 0 rawout
 0 bad vers, 23 bad scope, 0 bad options, 0 too many hdr
 0 no route, 0 can not forward, 0 redirect sent
 0 frag recv, 0 frag dropped, 0 frag timeout, 0 frag overflow
 0 reassembled, 0 fragmented, 0 ofragments, 0 can not frag
 0 too short, 0 too small, 11 not member
 0 no buffer, 66819 allocated, 21769 freed
 0 forward cache hit, 46 forward cache miss
ICMP6 Statistics
Received:
 0 dest unreachable, 0 pkt too big, 0 time exceeded, 0 param prob
 2 echo req, 1 echo reply, 0 mem query, 0 mem report, 0 mem red
 0 router soli, 2393 router adv, 106 nei soli, 3700 nei adv, 0 redirect
 0 bad code, 0 too short, 0 bad checksum, 0 bad len
 0 reflect, 0 nd toomany opt, 0 badhopcount
Sent:
 0 dest unreachable, 0 pkt too big, 0 time exceeded, 0 param prob
 1 echo req, 2 echo reply, 0 mem query, 0 mem report, 0 mem red
 0 router soli, 2423 router adv, 3754 nei soli, 102 nei adv, 0 redirect
 0 error, 0 can not send error, 0 too freq
Sent Errors:
 0 unreachable no route, 0 admin, 0 beyond scope, 0 address, 0 no port
 0 pkt too big, 0 time exceed transit, 0 time exceed reassembly
 0 param problem header, 0 nexthead, 0 option, 0 redirect, 0 unknown
UDP Statistics
 470 received, 7851 sent, 6 no port, 0 input errors
TCP Statistics
 57913 active opens, 0 passive opens, 57882 failed attempts
 159 active resets, 0 passive resets, 0 input errors
 565189 in segments, 618152 out segments, 171337 retransmission
```

Syntax: show ipv6 traffic

This show ipv6 traffic command displays the following information.

Field	Description
IPv6 statistics	
received	The total number of IPv6 packets received by the router.
sent	The total number of IPv6 packets originated and sent by the router.
forwarded	The total number of IPv6 packets received by the router and forwarded to other routers.
delivered	The total number of IPv6 packets delivered to the upper layer protocol.
rawout	This information is used by Brocade Technical Support.
bad vers	The number of IPv6 packets dropped by the router because the version number is not 6.
bad scope	The number of IPv6 packets dropped by the router because of a bad address scope.

Displaying global IPv6 information

Field	Description
bad options	The number of IPv6 packets dropped by the router because of bad options.
too many hdr	The number of IPv6 packets dropped by the router because the packets had too many headers.
no route	The number of IPv6 packets dropped by the router because there was no route.
can not forward	The number of IPv6 packets the router could not forward to another router.
redirect sent	This information is used by Brocade Technical Support.
frag rcv	The number of fragments received by the router.
frag dropped	The number of fragments dropped by the router.
frag timeout	The number of fragment timeouts that occurred.
frag overflow	The number of fragment overflows that occurred.
reassembled	The number of fragmented IPv6 packets that the router reassembled.
fragmented	The number of IPv6 packets fragmented by the router to accommodate the MTU of this router or of another device.
ofragments	The number of output fragments generated by the router.
can not frag	The number of IPv6 packets the router could not fragment.
too short	The number of IPv6 packets dropped because they are too short.
too small	The number of IPv6 packets dropped because they do not have enough data.
not member	The number of IPv6 packets dropped because the recipient is not a member of a multicast group.
no buffer	The number of IPv6 packets dropped because there is no buffer available.
forward cache miss	The number of IPv6 packets received for which there is no corresponding cache entry.
ICMP6 statistics	
Some ICMP statistics apply to both Received and Sent, some apply to Received only, some apply to Sent only, and some apply to Sent Errors only.	
Applies to received and sent	
dest unreachable	The number of Destination Unreachable messages sent or received by the router.
pkt too big	The number of Packet Too Big messages sent or received by the router.
time exceeded	The number of Time Exceeded messages sent or received by the router.
param prob	The number of Parameter Problem messages sent or received by the router.
echo req	The number of Echo Request messages sent or received by the router.
echo reply	The number of Echo Reply messages sent or received by the router.
mem query	The number of Group Membership Query messages sent or received by the router.
mem report	The number of Membership Report messages sent or received by the router.
mem red	The number of Membership Reduction messages sent or received by the router.
router soli	The number of Router Solicitation messages sent or received by the router.
router adv	The number of Router Advertisement messages sent or received by the router.

Field	Description
nei soli	The number of Neighbor Solicitation messages sent or received by the router.
nei adv	The number of Router Advertisement messages sent or received by the router.
redirect	The number of redirect messages sent or received by the router.
Applies to received only	
bad code	The number of Bad Code messages received by the router.
too short	The number of Too Short messages received by the router.
bad checksum	The number of Bad Checksum messages received by the router.
bad len	The number of Bad Length messages received by the router.
nd toomany opt	The number of Neighbor Discovery Too Many Options messages received by the router.
badhopcount	The number of Bad Hop Count messages received by the router.
Applies to sent only	
error	The number of Error messages sent by the router.
can not send error	The number of times the node encountered errors in ICMP error messages.
too freq	The number of times the node has exceeded the frequency of sending error messages.
Applies to sent errors only	
unreach no route	The number of Unreachable No Route errors sent by the router.
admin	The number of Admin errors sent by the router.
beyond scope	The number of Beyond Scope errors sent by the router.
address	The number of Address errors sent by the router.
no port	The number of No Port errors sent by the router.
pkt too big	The number of Packet Too Big errors sent by the router.
time exceed transit	The number of Time Exceed Transit errors sent by the router.
time exceed reassembly	The number of Time Exceed Reassembly errors sent by the router.
param problem header	The number of Parameter Problem Header errors sent by the router.
nextheader	The number of Next Header errors sent by the router.
option	The number of Option errors sent by the router.
redirect	The number of Redirect errors sent by the router.
unknown	The number of Unknown errors sent by the router.
UDP statistics	
received	The number of UDP packets received by the router.
sent	The number of UDP packets sent by the router.
no port	The number of UDP packets dropped because the packet did not contain a valid UDP port number.
input errors	This information is used by Brocade Technical Support.
TCP statistics	
active opens	The number of TCP connections opened by the router by sending a TCP SYN to another device.
passive opens	The number of TCP connections opened by the router in response to connection requests (TCP SYNs) received from other devices.
failed attempts	This information is used by Brocade Technical Support.

Clearing global IPv6 information

Field	Description
active resets	The number of TCP connections the router reset by sending a TCP RESET message to the device at the other end of the connection.
passive resets	The number of TCP connections the router reset because the device at the other end of the connection sent a TCP RESET message.
input errors	This information is used by Brocade Technical Support.
in segments	The number of TCP segments received by the router.
out segments	The number of TCP segments sent by the router.
retransmission	The number of segments that the router retransmitted because the retransmission timer for the segment had expired before the device at the other end of the connection had acknowledged receipt of the segment.

Clearing global IPv6 information

You can clear the following global IPv6 information:

- Entries from the IPv6 cache.
- Entries from the IPv6 neighbor table.
- IPv6 routes from the IPv6 route table.
- IPv6 traffic statistics.

Clearing the IPv6 cache

You can remove all entries from the IPv6 cache or specify an entry based on the following:

- IPv6 prefix.
- IPv6 address.
- Interface type.

For example, to remove entries for IPv6 address 2000:e0ff::1, enter the following command at the Privileged EXEC level or any of the Config levels of the CLI.

```
device#clear ipv6 cache 2000:e0ff::1
```

Syntax: `clear ipv6 cache [ipv6-prefix/prefix-length | ipv6-address | ethernet port | tunnel number | ve number | vrf vrf-name]`

You must specify the `ipv6-prefix` parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. 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.

You must specify the `ipv6-address` parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The `ethernet | tunnel | ve | vrf` parameter specifies the interfaces for which you can remove cache entries. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a VE, VRF, or tunnel interface, also specify the VE, VRF name, or tunnel number, respectively.

Clearing IPv6 neighbor information

You can remove all entries from the IPv6 neighbor table or specify an entry based on the following:

- IPv6 prefix

- IPv6 address
- Interface type

For example, to remove entries for Ethernet interface 1/3/1, enter the following command at the Privileged EXEC level or any of the CONFIG levels of the CLI.

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

Syntax: `clear ipv6 neighbor [ipv6-prefix / prefix-length | ipv6-address | ethernet port | ve number | vrf vrf-name]`

You must specify the ipv6-prefix parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. 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.

You must specify the ipv6-address parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **ethernet | ve | vrf** parameter specifies the interfaces for which you can remove cache entries. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a VRF or VE, also specify the VRF name or VE number respectively.

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.

For example, to clear IPv6 routes associated with the prefix 2000:7838::/32, enter the following command at the Privileged EXEC level or any of the Config levels of the CLI.

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

Syntax: `clear ipv6 route [ipv6-prefix/prefix-length | vrf vrf-name]`

The ipv6-prefix / prefix-length parameter clears routes associated with a particular IPv6 prefix. You must specify the ipv6-prefix parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. 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. If you specify a VRF parameter, specify the VRF name.

Clearing IPv6 traffic statistics

To clear all IPv6 traffic statistics (reset all fields to zero), enter the following command at the Privileged EXEC level or any of the Config levels of the CLI.

```
device(config)#clear ipv6 traffic
```

Syntax: `clear ipv6 traffic`

IPv4 Static Routing

• Overview of static routing.....	175
• Configuring a basic IP static route.....	176
• Adding metrics to a static route.....	177
• Naming an IP static route.....	178
• Removing a name or a static route.....	178
• Configuring a physical interface as next hop.....	179
• Configuring a virtual interface as next hop.....	179
• Configuring a tunnel as next hop.....	180
• Configuring a static route for use with a route map.....	180
• Configuring a null route.....	180
• Configuring a default static route.....	182
• Resolving a static route using other static routes.....	182
• Resolving the next hop through a protocol.....	183
• Creating an IP static route in a non-default VRF.....	183
• Configuring load sharing and redundancy.....	184
• Determining maximum static routes.....	186
• Displaying IPv4 static routes.....	187

Overview of static routing

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

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

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

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

- Destination network, using an IP address and network mask or prefix length
- Default network route
- Next-hop router
- Next-hop tunnel gateway
- Next-hop network protocol type
- 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.

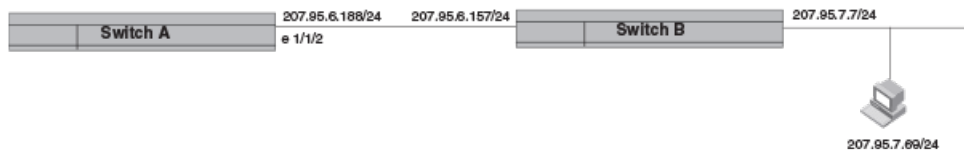
Static route states follow port states

IP static routes remain in the IP route table only as long as the port or virtual interface used by the route is available. 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.

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

FIGURE 14 Example of a static route



The following command configures a static route to 207.95.7.0, 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 1/1/2, 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 1/1/2.

The software automatically removes a static IP route from the IP route table if the port used by that route becomes unavailable. 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, or enter the IP address and network mask for the route destination network. On the same command line, enter the IP address for the next hop.

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

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

NOTE

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

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 255. 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
```

Naming an IP static route

IPv4 static route names are optional and non-unique. You can give a group of static routes the same name to help identify them.

To configure an IP static route with a name, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the destination IP address and mask or prefix length followed by the IP address of the next hop. On the same command line, enter the keyword **name** followed by the identifying ASCII string.

```
device(config)# ip route 10.22.22.22 255.255.255.0 10.1.1.1 name corporate
```

The following example creates a static route to IP destination network addresses beginning with 10.22.22.22 through the next-hop address 10.1.1.1. The route is given the non-unique name "corporate."

```
device# configure terminal
device(config)# ip route 10.22.22.22 255.255.255.0 10.1.1.1 name corporate
```

Removing a name or a static route

When an IP route has a name, the **no** form of the full **ip route** command removes the name. Use the **no** form of the command a second time to remove the route.

1. Enter configuration mode.

```
device# configure terminal
```

2. Enter **no ip route** followed by the full route designation.

```
device# configure terminal
device(config)# no ip route 10.22.22.22 255.255.255.255 10.1.1.1 name xyz
```

This example removes only the name of the route.

3. If necessary, repeat the **no ip route** command with the full route designation.

```
device(config)# no ip route 10.22.22.22 255.255.255.255 10.1.1.1
```

This example repeats the previous route. Because the route has no name, the command removes the designated static route.

The following example removes the name of the designated static route, removes the route, and saves the change to the IP routing table.

```
device# configure terminal
device(config)# no ip route 10.22.22.22 255.255.255.255 10.1.1.1 name xyz
device(config)# no ip route 10.22.22.22 255.255.255.255 10.1.1.1
```

Configuring a physical interface as next hop

The interface you use for the static route's next hop must have at least one IP address configured on it. The address does not need to be in the same subnet as the destination network.

NOTE

You cannot add an interface-based static route to a network if there is already a static route of any type with the same metric you specify for the interface-based route.

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, or enter the IP address and network mask for the route destination network. On the same command line, enter the keyword **ethernet** followed by the interface number to be used as next hop.

```
device(config)# ip route 10.128.2.69 255.255.255.0 ethernet 1/4/1
```

This example configures an IP static route with a destination network address of 10.128.2.69, a network mask of 255.255.255.0, and Ethernet port 1/4/1 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 1/2/1.

```
device# configure terminal
device(config)# ip route 10.0.0.0/24 ethernet 1/2/1
```

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 network mask or prefix-length. On the same command line, enter the keyword **ve** followed by the appropriate ID number.

```
device(config)# ip route 10.128.2.71 255.255.255.0 ve 3
```

The following example configures an IP static route with a destination address of 10.128.2.71, a prefix-length of /24, and a virtual interface (ve 3) as the next hop.

```
device# configure terminal
device(config)# ip route 10.128.2.71/24 ve 3
```

Configuring a tunnel as next hop

To configure an IP static route with a tunnel as the next hop, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Configure the destination IP address, followed by the prefix length or address mask. On the same command line, enter the keyword **tunnel** followed by the tunnel ID.

```
device(config)# ip route 10.128.2.71 255.255.255.0 tunnel 4
```

The following example configures an IP static route with a destination address of 10.128.2.71, a network mask of 255.255.255.0, and a tunnel gateway (tunnel 4) as the next hop.

```
device# configure terminal
device(config)# ip route 10.128.2.71 255.255.255.0 tunnel 4
```

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. 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
```

NOTE

An address mask may be used instead of the prefix-length (such as 255.255.255.0 instead of /24).

The following example creates an IP static route to destination IP addresses beginning with 10.0.0.0 through the next-hop address 10.1.1.1. The static route includes the tag "3" for later use in a route map.

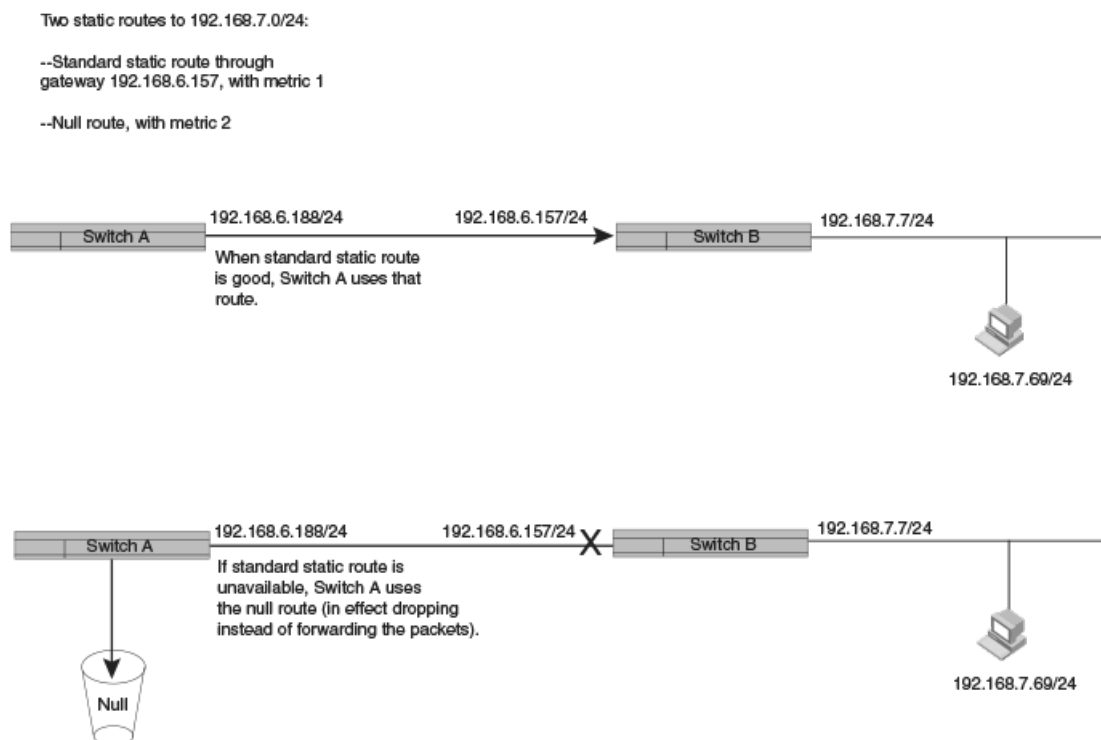
```
device# configure terminal
device(config)# ip route 10.0.0.0/24 10.1.1.1 tag 3
```

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 15 Null route and standard route to same destination



To configure a null route with a lower priority than the preferred route, perform the following steps.

1. **NOTE**

You cannot add a null static route to a network if there is already a static route of any type with the same metric you specify for the null route.

Enter global configuration mode.

```
device# configure terminal
```

2. 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.

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

```
device(config)# ip route 192.168.7.0/24 null0 2
```

This example creates a null static route to the same destination. The metric is set higher so that the preferred route is used if it is available. When the preferred route becomes unavailable, the null route is used, and traffic to the destination is dropped.

The following example creates a primary route to all destinations beginning with 192.168.7.0. It creates an alternative null route to drop the packets when the primary route is not available.

```
device# configure terminal
device(config)# ip route 192.168.7.0/24 192.168.6.157
device(config)# ip route 192.168.7.0/24 null0 2
```

Configuring a default static route

You can manually create a default static route that the router uses if there are no other default routes to a destination.

If the default route is a protocol route, that protocol needs to be enabled to resolve static routes. Use the **ip route next-hop** command to allow protocol resolution through the default route.

If the default route itself is a static route, you must configure the **ip route next-hop-enable-default** command to resolve other static routes through the default route. You may also configure recursive lookup to resolve the next hop.

Perform these steps to configure a default route.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the following destination route and network mask.

```
device(config)# ip route 0.0.0.0 0.0.0.0
```

3. On the same line, enter a valid next-hop address.

```
device(config)# ip route 0.0.0.0 0.0.0.0 10.24.4.1
```

The example creates a default route through IP address 10.24.4.1.

4. (Optional) Enable the default network route for static route resolution.

```
device(config)# ip route next-hop-enable-default
```

NOTE

This command can be independently applied on a per-VRF basis.

5. (Optional) Configure next-hop recursive lookup to resolve the next-hop gateway.

```
device# configure terminal
device(config)# ip route next-hop-recursion
```

The following example configures static routing next-hop recursion to three levels (the default). It configures the network default static route through next-hop IP address 10.24.4.1 and allows the default route to resolve other static routes.

NOTE

You can specify a level of recursion up to 10.

```
device# configure terminal
device(config)# ip route next-hop-recursion
device(config)# ip route 0.0.0.0 0.0.0.0 10.24.4.1
device(config)# ip route next-hop-enable-default
```

Resolving a static route using other static routes

You can use next-hop recursive lookup to resolve a static route.

Perform these steps to enable next-hop recursive lookup.

NOTE

This command can be independently applied on a per-VRF basis.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the following command, and, as an option, specify the level of recursion.

```
device(config)# ip route next-hop-recursion
```

This example configures recursive static route lookup to three levels (the default).

The following example configures recursive static route lookup to five levels.

```
device# configure terminal
device(config)# ip route next-hop-recursion 5
```

Resolving the next hop through a protocol

You can use routes from a specific protocol to resolve a configured static route.

Perform these steps to resolve the next hop for a static route using learned routes from a protocol.

NOTE

Connected routes are always used to resolve static routes.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable resolution through the desired protocol.

These protocol options are available:

- bgp
- ospf
- rip

```
device(config)# ip route next-hop ospf
```

This example enables route resolution through OSPF.

```
device(config)# ip route next-hop bgp
```

This example resolves static routes through BGP. Both iBGP and eBGP routes are used to resolve the routes.

```
device(config)# ip route next-hop rip
```

This example resolves static routes through RIP.

Creating an IP static route in a non-default VRF

You can configure an IP static route in a non-default VRF. If the VRF is not named, the default VRF is used.

The VRF configured must be a valid VRF.

To create an IP static route with a next hop in a non-default VRF, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **ip route** command followed by the keyword **vrf** and the VRF name. On the same command line, enter the destination IP address, followed by the prefix-length or the address mask and then the IP address of the next-hop.

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

This example configures an IP static route through the non-default VRF "blue" with the next-hop address 10.1.1.1.

The following example configures a static route with a destination address 56.1.5.0/24. The route is configured in the non-default VRF "red" and uses tunnel 5 as the next-hop gateway.

```
device# configure terminal
device(config)# ip route vrf red 56.1.5.0/24 tunnel 5
```

NOTE

When a tunnel is designated as the next-hop gateway for a non-default VRF destination, the tunnel must already exist before the static route can be created.

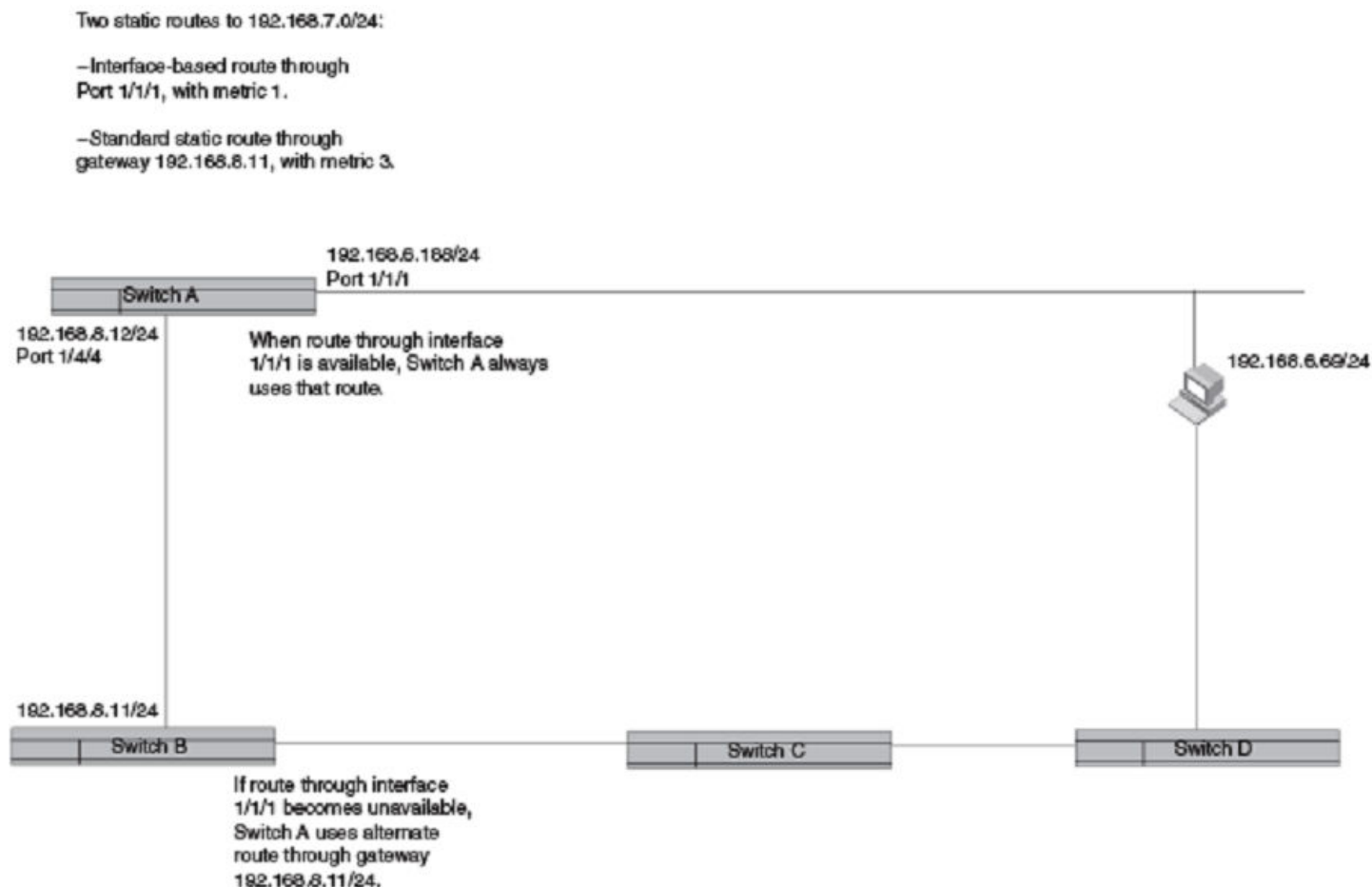
Configuring load sharing and redundancy

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 basic round-robin.

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 16 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 10.128.2.69/24 10.157.22.1
device(config)# ip route 10.128.2.69/24 10.111.10.1
device(config)# ip route 10.128.2.69/24 10.1.1.1
```

This example creates three next-hop gateways for all destinations beginning with 10.128.2.69. 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 10.128.2.69/24 10.157.22.1
device(config)# ip route 10.128.2.69/24 10.111.10.1 2
device(config)# ip route 10.128.2.69/24 10.1.1.1 3
```

This example creates three alternate routes to all destinations beginning with 10.128.2.69. 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.

Determining maximum static routes

You can modify the maximum number of static routes.

Perform these steps to check the maximum setting for static routes and to modify the value.

1. Verify the current maximum setting for the device.

```
device# show default values
```

The maximum number of static IP routes the system can retain is listed under System Parameters in the ip-static-route row as shown in the following example.

```
device# show default values
sys log buffers:50 mac age time:300 sec telnet sessions:5
ip arp age:10 min bootp relay max hops:4 ip ttl:64 hops
ip addr per intf:24
:
:
System Parameters      Default Maximum Current Configured
ip-arp                 4000    64000    4000    4000
ip-static-arp          512     6000    512     512
pim-mc-cache           1024    4096    1024    1024
:
:
ip-route                12000   15168   12000   12000
ip-static-route         64      2048    64      64
:
:
ip-vrf                  16      16      16      16
ip-route-default-vrf    12000   15168   12000   12000
ip6-route               5120    5120    5120    5120
ip6-route-default-vr    5120    5120    5120    5120
ip6-route-vrf           100     5120    100     100
device(config)#
```

2. Enter global configuration mode.

```
device# configure terminal
```

3. Specify a new maximum for IP static routes.

```
device(config)# system-max ip-static-route 4096
```

4. Save the change and reload the device.

```
device(config)# write memory
device(config)# reload
```

5. Verify the change with the **show default values** command.

The following example changes the system max value for IP static routes from 2048 to 4096.

```
device# configure terminal
device(config)#
device(config)# show default values
sys log buffers:50 mac age time:300 sec telnet sessions:5
ip arp age:10 min bootp relay max hops:4 ip ttl:64 hops
ip addr per intf:24
:
:
System Parameters      Default Maximum Current Configured
ip-arp                  4000    64000    4000    4000
ip-static-arp           512     6000     512     512
pim-mcache              1024    4096     1024    1024
:
:
ip-route                12000    15168    12000    12000
ip-static-route         64       2048     64       64
:
:
ip-vrf                  16       16       16       16
ip-route-default-vrf    12000    15168    12000    12000
ip6-route               5120     5120     5120     5120
ip6-route-default-vr    5120     5120     5120     5120
ip6-route-vrf           100      5120     100      100
device(config)#

device(config)# system-max ip-static-route 4096

device(config)# write memory
device(config)# reload

device(config)# show default values
sys log buffers:50 mac age time:300 sec telnet sessions:5
ip arp age:10 min bootp relay max hops:4 ip ttl:64 hops
ip addr per intf:24
:
:
System Parameters      Default Maximum Current Configured
ip-arp                  4000    64000    4000    4000
ip-static-arp           512     6000     512     512
pim-mcache              1024    4096     1024    1024
:
:
ip-route                12000    15168    12000    12000
ip-static-route         64       4096     64       64
:
:
ip-vrf                  16       16       16       16
ip-route-default-vrf    12000    15168    12000    12000
ip6-route               5120     5120     5120     5120
ip6-route-default-vr    5120     5120     5120     5120
ip6-route-vrf           100      5120     100      100
device(config)#
```

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 run ip route** command.
2. To display a list of active static routes and their connection times, at the device prompt, enter the **show ip route static** command.
3. To show all active IP routes and their connection times, enter the **show ip route** command.

Displaying IPv4 static routes

The following example shows two configured IPv4 routes from the management port. The first is the default route, 0.0.0.0/0. This is a static route ("S") that uses the next-hop gateway 10.25.224.1. The default route has a distance metric of 254 (beneath the threshold of 255, which would be unreachable) and a metric of 1. The route has been up for over 8 days.

The second route in this example is a directly connected route to all destinations beginning with 10.25.224.0. It has no extra costs associated with it and has been up for 6 hours and 39 minutes. The second route is the preferred route because, unlike the first route, it has no additional cost associated with it.

```
device# show ip route
Total number of IP routes: 2
Type Codes - B:BGP D:Connected O:OSPF R:RIP S:Static; Cost - Dist/Metric
BGP Codes - i:iBGP e:eBGP
OSPF Codes - i:Inter Area 1:External Type 1 2:External Type 2
```

	Destination	Gateway	Port	Cost	Type	Uptime
1	0.0.0.0/0	10.25.224.1	e mgmt1	254/1	S	8d6h
2	10.25.224.0/24	DIRECT	e mgmt1	0/0	D	6h39

IPv6 Static Routing

• Overview of static routing.....	189
• Configuring a basic IPv6 static route.....	190
• Removing an IPv6 static route.....	191
• Configuring an interface as next hop.....	191
• Configuring a virtual interface as next hop.....	192
• Configuring a tunnel as next hop.....	193
• Configuring a VRF as next hop for an IPv6 static route.....	193
• Adding metrics to an IPv6 static route.....	193
• Configuring a null route.....	194
• Configuring a default static route.....	196
• Resolving a static route using other static routes.....	196
• Resolving the IPv6 static route through a protocol.....	197
• Configuring load sharing and redundancy.....	197
• Adding an IPv6 static route tag for use with route-maps.....	199
• IPv6 multicast static routes.....	199
• Configuring IPv6 multicast routes in a non-default VRF.....	199
• Determining maximum static routes.....	201
• Displaying information on IPv6 static routes.....	203

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 and can use routing information learned from certain protocols.

Static routes are manually configured entries in the existing IPv6 routing table. In setting up static routes, you can specify several types of destinations:

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

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

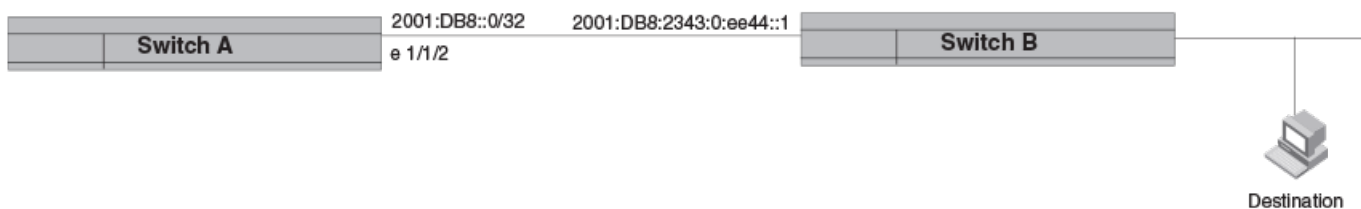
- Alternative routes to the same destination to help load balance traffic.

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 17 Example of a 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
```

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 1/1/2, 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 1/1/2.

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 the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

To configure a basic IPv6 static route, 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

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

The **no** form of the **ipv6 route** command must be entered with exact parameters to remove the command. If the route is configured in a non-default VRF, the **no** form of the **ipv6 route** command must be entered in VRF configuration mode.

Follow these steps to remove an IPv6 static route.

1. Enter global configuration mode.

```
device# configure terminal
```

2. (Optional) To view configured routes and confirm exact parameters, enter the command **show ipv6 route** to display the IPv6 route table.

```
device(config)# show ipv6 route
```

3. (Optional) Enter the **show ipv6 static route** command to narrow the output to static routes only.
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.

The following example removes an existing IPv6 static route from a non-default VRF.

```
device# configure terminal
device(config)# vrf corporate
device(config-vrf-corporate)# rd 20:10
device(config-vrf-corporate)# address-family ipv6
device(config-vrf-corporate-ipv6)# no ipv6 route 2002::/64 ethernet 1/1/1
```

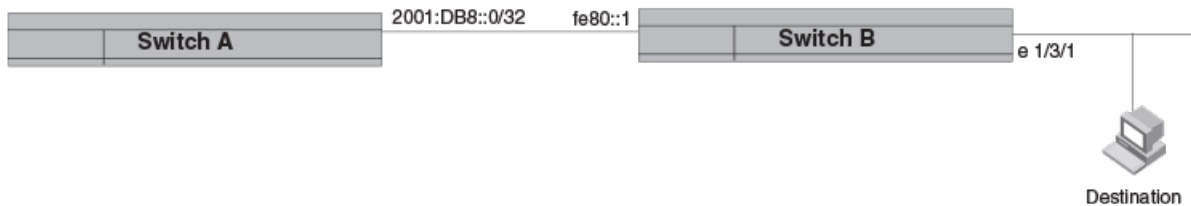
Configuring an interface as next hop

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

To configure an IPv6 static route with an interface as the next hop as depicted in the following illustration, perform these steps.

Configuring a virtual interface as next hop

FIGURE 18 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 keyword **ethernet** followed by the interface number as the next-hop, followed by its link-local IPv6 address.

```
device(config)# ipv6 route 2001:DB8::0/32 ethernet 1/3/1 fe80::1
```

The following example configures a static IPv6 route for a destination network with the prefix 2001:DB8::0/32 and a next-hop gateway with the link-local address fe80::1 that the Layer 3 switch can access through Ethernet interface 1/3/1.

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/32 ethernet 1/3/1 fe80::1
```

Configuring a virtual interface as next hop

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

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, add the keyword **ve** followed by the virtual interface ID to be used as the next hop, along with its link-local address.

```
device(config)# ipv6 route 2001:DB8::0/32 ve 3 fe80::1
```

This example shows the next-hop destination as virtual interface (ve) 3, with a link-local address of fe80::1.

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 ve 3 fe80::1
```


Configuring a tunnel as next hop

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

To configure a basic IPv6 static route through a next-hop tunnel, perform these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the IPv6 destination address and prefix, followed by the keyword **tunnel** and the tunnel ID.

```
device(config)# ipv6 route 2001:DB8::0/32 tunnel 1
```

The following example configures an IPv6 static route to 2001:DB8::0/32 destinations with a next-hop gateway through Tunnel 1.

```
device# configure terminal
device(config)# ipv6 route 2001:DB8::0/32 tunnel 1
```

Configuring a VRF as next hop for an IPv6 static route

A non-default VRF can be configured as the next-hop gateway for an IPv6 static route.

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

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 immediately by the keyword **vrf** and the name of the VRF that contains the next-hop router for the route. On the same command line, enter the destination IPv6 address, including the prefix length, and the IPv6 address of the next hop.

```
device(config)# ipv6 route vrf partners 2001:DB8::0/32 2001:DB8:0:ee44::1
```

This example creates an IPv6 static route for the destination network addresses with the prefix 2001:DB8::0/32 through the next-hop VRF named "partners" with the global IPv6 address 2001:DB8:0:ee44::1.

Adding metrics to an IPv6 static route

You can influence how likely a static route is to be used by modifying the cost metric or the administrative distance.

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

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 255. The default is 1. A value of 255 is considered unreachable.

```
device(config)# ipv6 route 2001:DB8::0/32 2001:DB8:2343:0:ee44::1 2
```

This example configures a static IPv6 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 and assigns the route a metric of 2.

```
device(config)# ipv6 route 2001:DB8::0/32 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::0/32 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::0/32 2001:DB8:0:ee44::1 distance 3
```

Configuring a null route

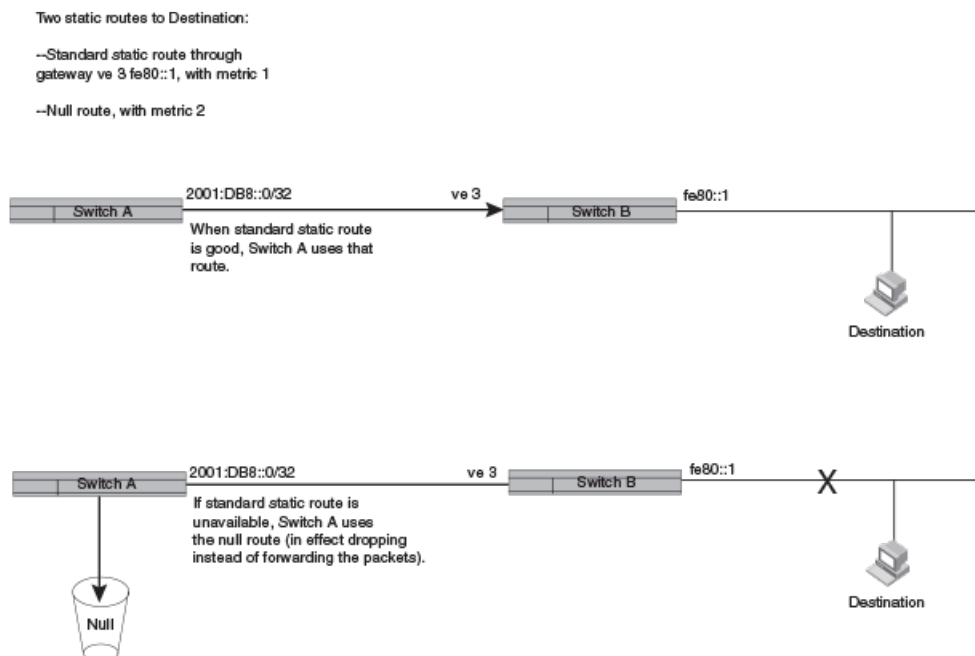
You can configure a null static route to drop packets to a certain destination. This is useful when the traffic should not be forwarded if the preferred route is unavailable.

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 19 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::0/32 ve 3 fe80::1
```

This example creates a static route to IPv6 2001 : DB8 : : 0/32 destination addresses. These destinations are routed through link-local address fe80::1 and the next hop gateway virtual interface (ve) 3.

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

```
device(config)# ipv6 route 2001:DB8::0/32 null0 2
```

This example creates a null static route to the same destination. The metric is set higher so that the preferred route is used if it is available.

The following example creates a primary route to all 2001 : DB8 : : 0/32 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 : : 0/32 ve 3 fe80::1
device(config)# ipv6 route 2001 : DB8 : : 0/32 null0 2
```

Configuring a default static route

You can manually create a default static route that the router uses if there are no other default routes to a destination.

Because the default route is a static route, you must configure the **ip route next-hop-enable-default** command to resolve other static routes through the default route.

Perform these steps to configure a default route.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable the default network route for static route resolution of routes to a particular destination.

```
device(config)# ipv6 route next-hop-enable-default
```

This example enables the default static route to resolve the next hop for IPv6 static routes.

NOTE

This command can be independently applied on a per-VRF basis.

3. (Optional) Configure the default route for recursive lookup of the next-hop.

```
device# configure terminal
device(config)# ipv6 route next-hop-recursion
```

This example allows three levels of recursion in looking up the next hop for any IPv6 static route. The default is 3. You may enter any value from 1 to 10.

4. 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. The route is able to resolve static routes using next-hop recursion to three levels (the default).

```
device# configure terminal
device(config)# ipv6 route next-hop-enable-default
device(config)# ipv6 route next-hop-recursion
device(config)# ipv6 route ::/0 2001:DB8:0:ee44::1
```

Resolving a static route using other static routes

You can use other static routes to resolve a static route through recursive lookup in local routing tables, up to 10 hops away.

Perform these steps to enable next-hop recursive lookup.

NOTE

This command can be independently applied on a per-VRF basis.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the following command, and, as an option, specify the level of recursion. You can enter a value from 1 through 10. If no value is specified, the level of recursion is 3.

```
device(config)# ipv6 route next-hop-recursion
```

This example configures recursive static route lookup to three levels (the default).

The following example configures recursive static route lookup to five levels.

```
device# configure terminal
device(config)# ipv6 route next-hop-recursion 5
```

Resolving the IPv6 static route through a protocol

You can use routes from another protocol to resolve a static route.

Perform these steps to resolve the next hop for an IPv6 static route using learned routes from BGP, OSPF, or RIP protocol.

NOTE

Connected routes are always used to resolve static routes.

1. Enter global configuration mode.

```
device# configure terminal
```

2. (Optional) Designate the non-default VRF of the destination.
3. Designate next-hop resolution through BGP, RIP, or OSPF protocol.

```
device(config)# ipv6 route next-hop vrf blue bgp
```

This example designates that the VRF named "blue" is to be used and that static routes are to be resolved through BGP.

The following example specifies that IPv6 static routes can be resolved through directly connected OSPF routers (instead of link-local IPv6 route tables, for example).

```
device# configure terminal
device(config)# ipv6 route next-hop ospf
```

Configuring load sharing and redundancy

You can configure multiple IP static routes to the same destination to set up load sharing or backup routes.

If you configure more than one static route to the same destination with different next-hop gateways but the same metrics, the router load balances among the routes using basic round-robin.

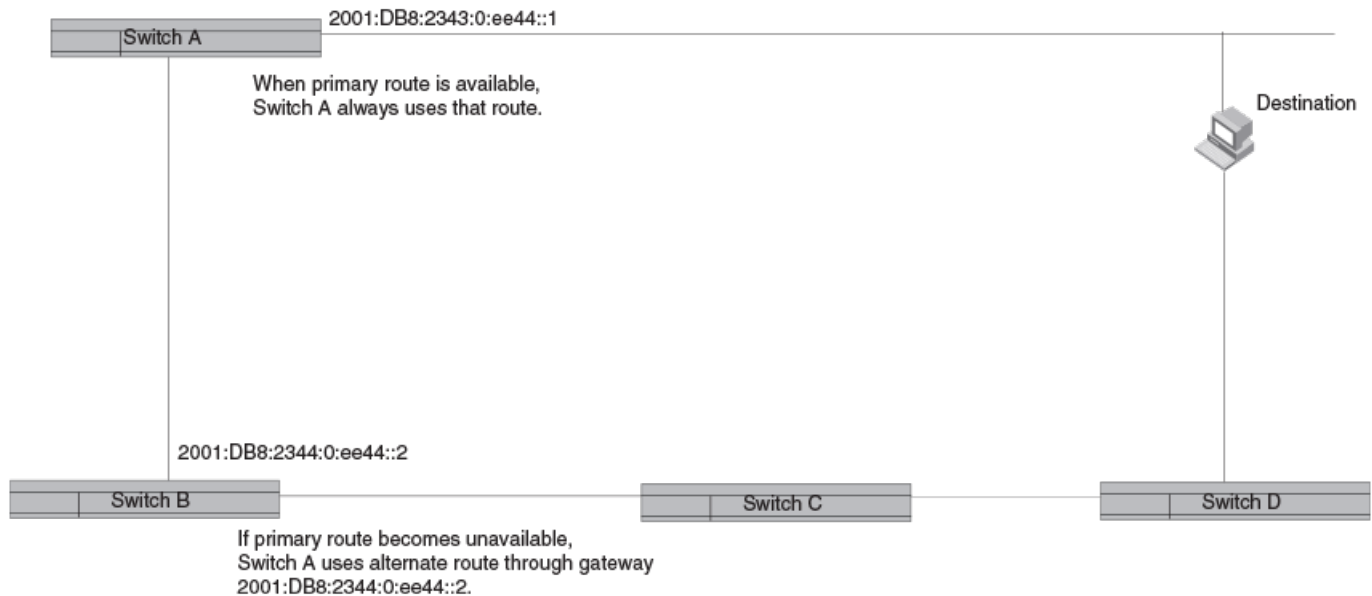
If you configure multiple static IP routes to the same destination with different next-hop gateways and different metrics, the router always uses the route with the lowest metric. If this route becomes unavailable, the router fails over to the static route with the next-lowest metric. The following figure depicts multiple routes with different metrics configured for the same destination.

FIGURE 20 Two static routes to same destination

Two static routes to 2001:DB8::0/32:

--Primary static route through gateway 2001:1:DB8:2343:0:ee44::1, with default metric 1.

--Standard static route through gateway 2001:DB8:2344:0:ee44::2, with metric 2.



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::0/32 2001:DB8:2343:0:ee44::1
device(config)# ipv6 route 2001:DB8::0/32 2001:DB8:2344:0:ee44::2
```

This example creates two next-hop gateways for all 2001:DB8::0/32 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::0/32 2001:DB8:2343:0:ee44::1
device(config)# ipv6 route 2001:DB8::0/32 2001:DB8:2344:0:ee44::2 2
```

This example creates an alternate route to all 2001:DB8::0/32 destinations. The primary route uses 2001:DB8:2343:0:ee44::1 as the next hop. The route has the default metric of 1. If this path is not available, traffic is directed through 2001:DB8:2344:0:ee44::2, which has the next lowest metric (2).

Adding an IPv6 static route tag for use with route-maps

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the Layer 3 switch using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

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.

```
ipv6 route 2001:DB8::0/32 2001:DB8:0:ee44::1 tag 3
```

The following example configures an IPv6 route to IPv6 2001:DB8::0/32 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::0/32 2001:DB8:0:ee44::1 tag 3
```

IPv6 multicast static routes

IPv6 multicast routes allow you to control the network path used by multicast traffic.

Static multicast routes are especially useful when the unicast and multicast topologies of a network are different. You can avoid the need to make the topologies similar by instead configuring static multicast routes.

You can configure more than one static IPv6 multicast route. The Brocade device by default uses the most specific route that matches a multicast source address. Thus, if you want to configure a multicast static route for a specific multicast source and also configure another multicast static route for all other sources, you can configure two static routes.

You can also influence route preference using cost metrics and administrative distance parameters.

NOTE

Regardless of the administrative distances, the Brocade device always prefers directly connected routes over other routes.

The following example configures an IPv6 multicast static route for a destination network with the prefix 2001:db8::0/32, a next-hop gateway with the global address 2001:db8:0:ee44::1, and an administrative distance of 110.

```
device# configure terminal
device(config)# ipv6 mroute 2001:db8::0/32 2001:db8:0:ee44::1 distance 110
```

Configuring IPv6 multicast routes in a non-default VRF

You can specify a default or non-default VRF for an IPv6 multicast static route. If no VRF is specified, the default VRF is used.

The VRF specified must be an existing VRF.

Follow these steps to create an IPv6 multicast static route in a non-default VRF.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Designate the non-default VRF for the IPv6 multicast static route.

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

This example configures "corporate" as the non-default VRF.

3. Give the route a Route Descriptor.

```
device(config-vrf-corporate)# rd 20:10
```

This example configures 20:10 as the route descriptor.

4. Specify the address family as IPv6.

```
device(config-vrf-corporate)# address-family ipv6
```

5. Configure the IPv6 static route, including destination IP address, mask prefix, and next-hop information.

```
device(config-vrf-corporate-ipv6)# ipv6 mroute 2002::/64 ethernet 1/1/1
```

This example configures an IPv6 static route to IP address 2002::/64 destinations via next-hop interface 1/1/1.

The following example creates an IPv6 multicast route with the RD 20:10 to 2002::/64 IP address via next-hop Ethernet interface 1/1/1. The route has a cost metric of 5.

```
device# configure terminal
device(config)# vrf corporate
device(config-vrf-corporate)# rd 20:10
device(config-vrf-corporate)# address-family ipv6
device(config-vrf-corporate-ipv6)# ipv6 mroute 2002::/64 ethernet 1/1/1 5
```


Determining maximum static routes

You can modify the maximum number of static routes.

Perform these steps to check the maximum setting for static routes and to modify the value.

1. Verify the current maximum setting for the device.

```
device# show default values
```

The maximum number of static IPv6 routes the system can retain is listed under System Parameters in the ip6-route row as shown in the following example.

```
device# show default values
sys log buffers:50 mac age time:300 sec telnet sessions:5
ip arp age:10 min bootp relay max hops:4 ip ttl:64 hops
ip addr per intf:24
:
:
System Parameters      Default Maximum Current Configured
ip-arp                  4000    64000    4000    4000
ip-static-arp           512     6000     512     512
pim-mcachel            1024    4096     1024    1024
:
:
ip-route                12000   15168    12000   12000
ip-static-route         64      2048     64      64
:
:
ip-vrf                  16      16       16      16
ip-route-default-vrf    12000   15168    12000   12000
ip6-route               5120    5120     5120    5120
ip6-route-default-vr    5120    5120     5120    5120
ip6-route-vrf           100     5120     100     100
device(config)#
```

2. Enter global configuration mode.

```
device# configure terminal
```

3. Specify a new maximum for IP static routes.

```
device(config)# system-max ip6-route 9000
```

4. Save the change and reload the device.

```
device(config)# write memory
device(config)# reload
```

5. Verify the change with the **show default values** command.

Determining maximum static routes

The following example changes the system max value for IP static routes from 5120 to 9000.

```

device# configure terminal
device(config)# show default values
sys log buffers:50 mac age time:300 sec telnet sessions:5
ip arp age:10 min bootp relay max hops:4 ip ttl:64 hops
ip addr per intf:24
:
:
System Parameters      Default Maximum Current Configured
ip-arp                 4000    64000    4000    4000
ip-static-arp          512     6000     512     512
pim-mcache             1024    4096     1024    1024
:
:
ip-route               12000    15168    12000    12000
ip-static-route        64       2048     64       64
:
:
ip-vrf                 16       16       16       16
ip-route-default-vrf  12000    15168    12000    12000
ip6-route              5120    5120     5120     5120
ip6-route-default-vr  5120    5120     5120     5120
ip6-route-vrf          100     5120     100     100
device(config)#

device(config)# system-max ip6-route 9000

device(config)# write memory
device(config)# reload

device(config)# show default values
sys log buffers:50 mac age time:300 sec telnet sessions:5
ip arp age:10 min bootp relay max hops:4 ip ttl:64 hops
ip addr per intf:24
:
:
System Parameters      Default Maximum Current Configured
ip-arp                 4000    6400    4000    4000
ip-static-arp          512     6000     512     512
pim-mcache             1024    4096     1024    1024
:
:
ip-route               12000    15168    12000    12000
ip-static-route        64       4096     64       64
:
:
ip-vrf                 16       16       16       16
ip-route-default-vrf  12000    15168    12000    12000
ip6-route              5120    9000     5120     5120
ip6-route-default-vr  5120    5120     5120     5120
ip6-route-vrf          100     5120     100     100
device(config)#

```

Displaying information on IPv6 static routes

You can consult the IPv6 route table for information on connected, static, and protocol routes.

To display information on IPv6 static routes, use the following commands.

1. To check whether IPv6 is enabled, enter the **show ipv6** command. The command can be entered at the device prompt or in global or interface configuration mode.

```
device# show ipv6
Global Settings
IPv6 is enabled
Link-local address(es):
fe80::ce4e:24ff:fe8b:b050 [Preferred]
Global unicast address(es):
Joined group address(es):
ff02::1:ff8b:b050
ff02::1
MTU is 1500 bytes
ND DAD is enabled, number of DAD attempts: 3
ND reachable time is 30000 milliseconds
ND retransmit interval is 1000 milliseconds
Current Hop Limit is 64
Hosts use stateless autoconfig for addresses
No Inbound Access List Set
No Outbound Access List Set
No IPv6 Domain Name Set
No IPv6 DNS Server Address set
```

This example shows IPv6 is enabled on the device. The preferred link-local address for the device is also shown.

2. To display the IPv6 route table information, enter the **show ipv6 route** command at the device prompt.

```
device# show ipv6 route
IPv6 Routing Table - 7 entries:
Type Codes: C - Connected, S - Static, R - RIP, O - OSPF, B - BGP
OSPF Sub Type Codes: O - Intra, Oi - Inter, Ol - Type1 external, O2 - Type2 external
Type      IPv6 Prefix      Next Hop Router      Interface      Dis/Metric
C         2000:4::/64       ::                   ethe 1/3/2     0/0
S         2001:DB8::/16     ::                   tunnel 6       1/1
S         2001:DB8:1234::/32 ::                   tunnel 6       1/1
C         2001:DB8:46a::/64  ::                   ethe 1/3/2     0/0
C         2001:DB8::1/128  ::                   loopback 2     0/0
O         2001:DB8::2/128   fe80::2e0:52ff:fe91:bb37 ethe 1/3/2     110/1
C         2001:DB8::/64    ::                   tunnel 2       0/0
```

As shown in the example, 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.

RIP

• RIP overview.....	205
• RIP parameters and defaults.....	205
• Configuring RIP parameters.....	207
• Displaying RIP Information.....	214
• Displaying CPU utilization statistics.....	216

RIP overview

Routing Information Protocol (RIP) is an IP route exchange protocol that uses a distance vector (a number representing distance) to measure the cost of a given route. The cost is a distance vector because the cost often is equivalent to the number of router hops between the Brocade device and the destination network.

A Brocade device can receive multiple paths to a destination. The software evaluates the paths, selects the best path, and saves the path in the IP route table as the route to the destination. Typically, the best path is the path with the fewest hops. A hop is another router through which packets must travel to reach the destination. If a RIP update is received from another router that contains a path with fewer hops than the path stored in the Brocade device route table, the older route is replaced with the newer one. The new path is then included in the updates sent to other RIP routers, including Brocade devices.

RIP routers, including Brocade devices, also can modify a route cost, generally by adding to it, to bias the selection of a route for a given destination. In this case, the actual number of router hops may be the same, but the route has an administratively higher cost and is thus less likely to be used than other, lower-cost routes.

A RIP route can have a maximum cost of 15. Any destination with a higher cost is considered unreachable. Although limiting to larger networks, the low maximum hop count prevents endless loops in the network.

Brocade devices support the following RIP versions:

- Version 1 (v1)
- Version 2 (v2, the default)
- V1 compatible with v2

RIP parameters and defaults

You can configure global RIP parameters for the protocol and interface RIP parameters on those interfaces that send and receive RIP information.

RIP global parameters

TABLE 39 RIP global parameters

Parameter	Description	Default
RIP state	The global state of the protocol.	Disabled

TABLE 39 RIP global parameters (continued)

Parameter	Description	Default
	NOTE You also must enable the protocol on individual interfaces. Globally enabling the protocol does not allow interfaces to send and receive RIP information.	
Administrative distance	<p>The administrative distance is a numeric value assigned to each type of route on the device.</p> <p>When the device is selecting from among multiple routes (sometimes of different origins) to the same destination, the device compares the administrative distances of the routes and selects the route with the lowest administrative distance.</p> <p>This parameter applies to routes originated by RIP. The administrative distance stays with a route when it is redistributed into other routing protocols.</p>	120
Redistribution	RIP can redistribute routes from other routing protocols such as OSPF and BGP4 into RIP. A redistributed route is one that a router learns through another protocol, and then distributes into RIP.	Disabled
Redistribution metric	<p>RIP assigns a RIP metric (cost) to each external route redistributed from another routing protocol into RIP.</p> <p>An external route is a route with at least one hop (packets must travel through at least one other router to reach the destination). This parameter applies to routes that are redistributed from other protocols into RIP.</p>	1
Update Interval	How often the router sends route updates to its RIP neighbors.	30 seconds
Learning default routes	<p>The device can learn default routes from its RIP neighbors.</p> <p>NOTE You also can enable or disable this parameter on an individual interface basis.</p>	Disabled
Advertising and learning with specific neighbors	The device learns and advertises RIP routes with all its neighbors by default. You can prevent the device from advertising routes to specific neighbors or learning routes from specific neighbors.	Learning and advertising permitted for all neighbors

RIP interface parameters

TABLE 40 RIP interface parameters

Parameter	Description	Default
RIP state and version	<p>The state of the protocol and the version that is supported on the interface. The version can be one of the following:</p> <ul style="list-style-type: none"> Version 1 only Version 2 only Version 1, but also compatible with version 2 <p>NOTE You also must enable RIP globally.</p>	Disabled
Metric	A numeric cost the device adds to RIP routes learned on the interface. This parameter applies only to RIP routes.	1
Learning default routes	Locally overrides the global setting.	Disabled
Loop prevention	<p>The method a device uses to prevent routing loops caused by advertising a route on the same interface as the one on which the device learned the route.</p> <ul style="list-style-type: none"> Split horizon - The device does not advertise a route on the same interface as the one on which the device learned the route. Poison reverse - The device assigns a cost of 16 ("infinite" or "unreachable") to a route before advertising it on the same interface as the one on which the device learned the route. <p>NOTE Enabling poison reverse disables split horizon on the interface.</p>	Split horizon
Advertising and learning specific routes	You can control the routes that a device learns or advertises.	The device learns and advertises all RIP routes on all interfaces.

Configuring RIP parameters

Enabling RIP

RIP is disabled by default. To enable RIP, you must enable it globally and also on individual interfaces on which you want to advertise RIP. Globally enabling the protocol does not enable it on individual interfaces. When you enable RIP on a port, you also must specify the version (version 1 only, version 2 only, or version 1 compatible with version 2).

To enable RIP globally, enter the **router rip** command.

```
device(config)# router rip
```

Syntax: [no] router rip

After globally enabling the protocol, you must enable it on individual interfaces. You can enable the protocol on physical interfaces as well as virtual routing interfaces. To enable RIP on an interface, enter commands such as the following.

```
device(config)# interface ethernet 1/1/1
device(config-if-e1000-1/1/1)# ip rip v1-only
```

Syntax: [no] ip rip {v1-only | v1-compatible-v2 | v2-only}

Configuring route costs

By default, a Brocade device port increases the cost of a RIP route that is learned on the port. The Brocade device increases the cost by adding one to the route metric before storing the route.

You can change the amount that an individual port adds to the metric of RIP routes learned on the port.

To increase the metric for learned routes, enter the **ip rip metric-offset** command.

```
device(config-if-e1000-1/1/1)# ip rip metric-offset 5 in
```

In the above example, the **ip rip metric-offset** command configures the port to add 5 to the cost of each route it learns.

Syntax: [no] ip rip metric-offset *num* {in | out}

The *num* variable specifies a range from 1 through 16.

NOTE

RIP considers a route with a metric of 16 to be unreachable. You can prevent the device from using a specific port for routes learned through that port by setting its metric to 16.

The **in** keyword applies to routes the port learns from RIP neighbors.

The **out** keyword applies to routes the port advertises to its RIP neighbors.

Changing the administrative distance

By default, the Brocade device assigns the default RIP administrative distance (120) to RIP routes. When comparing routes based on administrative distance, the Brocade device selects the route with the lower distance. You can change the administrative distance for RIP routes.

To change the administrative distance for RIP routes, enter the **distance** command.

```
device(config-rip-router)# distance 140
```

In the above example, the **distance** command changes the administrative distance to 140 for all RIP routes.

Syntax: [no] distance *number*

The *number* variable specifies a range from 1 through 255.

Configuring redistribution

You can configure the Brocade device to redistribute routes learned through Open Shortest Path First (OSPF) or Border Gateway Protocol version 4 (BGP4), connected into RIP, or static routes. When you redistribute a route from one of these other protocols into RIP, the Brocade device can use RIP to advertise the route to its RIP neighbors.

To configure redistribution, perform the following tasks.

1. Configure redistribution filters (optional). You can configure filters to permit or deny redistribution for a route based on its origin (OSPF, BGP4, and so on), the destination network address, and the route's metric. You also can configure a filter to set the metric based on these criteria.
2. Change the default redistribution metric (optional). The Brocade device assigns a RIP metric of 1 to each redistributed route by default. You can change the default metric to a value up to 15.
3. Enable redistribution.

NOTE

Do not enable redistribution until you configure the other redistribution parameters.

Configuring redistribution filters

RIP redistribution filters apply to all interfaces. Use route maps to define how you want to deny or permit redistribution.

NOTE

The default redistribution action is permit, even after you configure and apply redistribution filters to the virtual routing interface. If you want to tightly control redistribution, apply a filter to deny all routes as the last filter (the filter with the highest ID), and then apply filters to allow specific routes.

A route map is a named set of match conditions and parameter settings that the Brocade 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 up to 50 instances. The Brocade device evaluates a route according to a route map's instances in ascending numerical order. The route is first compared against instance 1, then against instance 2, and so on. If a match is found, the Brocade device stops evaluating the route against the route map instances.

Route maps can contain match statements and set statements. Each route map contains a permit or deny action for routes that match the match statements:

- If the route map contains a permit action, a route that matches a match statement is permitted; otherwise, the route is denied.
- If the route map contains a deny action, a route that matches a match statement is denied.
- If a route does not match any match statements in the route map, 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 route is considered to be a match.
- For route maps that contain address filters, AS-path filters, or community filters, if the action specified by a filter conflicts with the action specified by the route map, the route map's action takes precedence over the individual filter's action.

If the route map contains set statements, routes that are permitted by the route map's match statements are modified according to the set statements.

In RIP, the match statements are based on prefix lists and access control lists. Set statements are based on tag values and metric values.

To configure redistribution filters, enter the following command.

```
device(config-rip-router)# redistribute connected route-map routemap1
```

Syntax: [no] redistribute {connected | bgp | ospf | static [metric value | route-map name]}

The **connected** keyword applies redistribution to connected types.

The **bgp** keyword applies redistribution to BGP4 routes.

The **ospf** keyword applies redistribution to OSPF routes.

The **static** keyword applies redistribution to IP static routes.

The **metric value** parameter sets the RIP metric value from 1 through 15 that will be applied to the routes imported into RIP.

The **route-map name** parameter indicates the route map's name.

Matching based on RIP protocol type

The **match** option has been added to the **route-map** command that allows statically configured routes or the routes learned from the IGP protocol RIP.

To configure the route map to match to RIP, enter the **match protocol rip** command.

```
device(config-routemap test)# match protocol rip
```

Syntax: [no] match protocol rip

Changing the default redistribution metric

When the Brocade device redistributes a route into RIP, the software assigns a RIP metric (cost) to the route. By default, the software assigns a metric of 1 to each route that is redistributed into RIP. You can increase the metric that the Brocade device assigns, up to 15.

To change the RIP metric the Brocade device assigns to redistributed routes, enter a command such as the following.

```
device(config-rip-router)# default-metric 10
```

This command assigns a RIP metric of 10 to each route that is redistributed into RIP.

Syntax: [no] default-metric 1-15

Configuring route learning and advertising parameters

By default, a Brocade device learns routes from all its RIP neighbors and advertises RIP routes to those neighbors.

You can configure the following learning and advertising parameters:

- Update interval - The update interval specifies how often the device sends RIP route advertisements to its neighbors. You can change the interval to a value from 3 through 65535 seconds. The default is 30 seconds.
- Learning and advertising of RIP default routes - The Brocade device can learn and advertise RIP default routes. You can disable learning and advertising of default routes on a global or individual interface basis.
- Learning of standard RIP routes - By default, the Brocade device can learn RIP routes from all its RIP neighbors. You can configure RIP neighbor filters to explicitly permit or deny learning from specific neighbors.

Changing the update interval for route advertisements

The update interval specifies how often the device sends route advertisements to its RIP neighbors. You can specify an interval from 3 through 21,845 seconds. The default is 30 seconds.

To change the RIP update interval, enter the **update-time** command.

```
device(config-rip-router)# update-time 120
```

This command configures the device to send RIP updates every 120 seconds.

Syntax: update-time value

Enabling learning of RIP default routes

By default, the Brocade device does not learn default RIP routes. You can enable learning of RIP default routes on a global or interface basis.

To enable learning of default RIP routes on a global basis, enter the following command.

```
device(config-rip-router)# learn-default
```

Syntax: [no] learn-default

To enable learning of default RIP routes on an interface, enter the ip rip learn-default command.

```
device(config)# interface ethernet 1/1/1
device(config-if-e10000-1/1/1)# ip rip learn-default
```

Syntax: [no] ip rip learn-default

Configuring a RIP neighbor filter

By default, a Brocade device learns RIP routes from all its RIP neighbors. Neighbor filters allow you to specify the neighbor routers from which the Brocade device can receive RIP routes. Neighbor filters apply globally to all ports.

To configure a RIP neighbor filters, enter the **neighbor** command.

```
device(config-rip-router)# neighbor 1 deny any
```

This command configures the Brocade device so that the device does not learn any RIP routes from any RIP neighbors.

Syntax: [no] neighbor filter-num {permit | deny} {source-ip-address | any}

The following commands configure the Brocade device to learn routes from all neighbors except 10.70.12.104. Once you define a RIP neighbor filter, the default action changes from learning all routes from all neighbors to denying all routes from all neighbors except the ones you explicitly permit. Thus, to deny learning from a specific neighbor but allow all other neighbors, you must add a filter that allows learning from all neighbors. Make sure you add the filter to permit all neighbors as the last filter (the one with the highest filter number). Otherwise, the software can match on the permit all filter before a filter that denies a specific neighbor, and learn routes from that neighbor.

```
device(config-rip-router)# neighbor 2 deny 10.70.12.104
device(config-rip-router)# neighbor 64 permit any
```

Changing the route loop prevention method

RIP uses the following methods to prevent routing loops:

- Split horizon - The device does not advertise a route on the same interface as the one on which the Brocade device learned the route. This is the default.
- Poison reverse - The device assigns a cost of 16 ("infinite" or "unreachable") to a route before advertising it on the same interface as the one on which the Brocade device learned the route.

These loop prevention methods are configurable on a global basis as well as on an individual interface basis. One of the methods is always in effect on an interface enabled for RIP. Thus, if you disable one method, the other method is enabled.

NOTE

These methods are in addition to RIP's maximum valid route cost of 15.

To disable poison reverse and enable split horizon on a global basis, enter the following command.

```
device(config-rip-router)# no poison-reverse
```

Syntax: [no] poison-reverse

To disable poison reverse and enable split horizon on an interface, enter commands such as the following.

```
device(config)#interface ethernet 1/1/1
device(config-if-e10000-1/1/1)# no ip rip poison-reverse
```

Syntax: [no] ip rip poison-reverse

To disable split horizon and enable poison reverse on an interface, enter commands such as the following.

```
device(config)#interface ethernet 1/1/1
device(config-if-e10000-1/1/1)# ip rip poison-reverse
```

You can configure the Brocade device to avoid routing loops by advertising local RIP routes with a cost of 16 ("infinite" or "unreachable") when these routes go down.

```
device(config-rip-router)# poison-local-routes
```

Syntax: [no] poison-local-routes

Suppressing RIP route advertisement on a VRRP or VRRPE backup interface

NOTE

This section applies only if you configure the device for Virtual Router Redundancy Protocol (VRRP) or VRRP Extended (VRRPE).

Normally, a VRRP or VRRPE Backup includes route information for the virtual IP address (the backed up interface) in RIP advertisements. As a result, other routers receive multiple paths for the backed up interface and might sometimes unsuccessfully use the path to the Backup rather than the path to the Master.

You can prevent the backups from advertising route information for the backed up interface by enabling suppression of the advertisements.

To suppress RIP advertisements for the backed up interface, enter the following commands.

```
device(config)# router rip
device(config-rip-router)# use-vrrp-path
```

Syntax: [no] use-vrrp-path

The syntax is the same for VRRP and VRRP-E.

Configuring RIP route filters using prefix-lists and route maps

You can configure prefix lists to permit or deny specific routes, then apply them globally or to individual interfaces and specify whether the lists apply to learned routes (in) or advertised routes (out).

You can configure route maps to permit or deny specific routes, then apply a route map to an interface, and specify whether the map applies to learned routes (in) or advertised routes (out).

NOTE

A route is defined by the destination's IP address and network mask.

NOTE

By default, routes that do not match a prefix list are learned or advertised. To prevent a route from being learned or advertised, you must configure a prefix list to deny the route.

To configure a prefix list, enter commands such as the following.

```
device(config)# ip prefix-list list1 permit 10.53.4.1 255.255.255.0
device(config)# ip prefix-list list2 permit 10.53.5.1 255.255.255.0
device(config)# ip prefix-list list3 permit 10.53.6.1 255.255.255.0
device(config)# ip prefix-list list4 deny 10.53.7.1 255.255.255.0
```

The prefix lists permit routes to three networks, and deny the route to one network.

Because the default action is permit, all other routes (routes not explicitly permitted or denied by the filters) can be learned or advertised.

Syntax: [no] ip prefix-list *name* {permit | deny} {*source-ip-address* | any *source-mask* | any}

To apply a prefix list at the global level of RIP, enter commands such as the following.

```
device(config-rip-router)# prefix-list list1 in
```

Syntax: no prefix-list *name* {in | out}

To apply prefix lists to a RIP interface, enter commands such as the following.

```
device(config-if-e1000-1/1/2)# ip rip prefix-list list2 in
device(config-if-e1000-1/1/2)# ip rip prefix-list list3 out
```

Syntax: no ip rip prefix-list *name* {in | out}

In is for Inbound filtering. It applies the prefix list to routes the Brocade device learns from its neighbor on the interface.

Out is for Outbound filtering. It applies the prefix list to routes the Brocade device advertises to its neighbor on the interface.

The commands apply RIP list2 route filters to all routes learned from the RIP neighbor on the port and applies the lists to all routes advertised on the port.

To configure a route-map, enter commands such as the following.

```
device(config)#access-list 21 deny 160.1.0.0 0.0.255.255
device(config)#access-list 21 permit any
device(config)# route-map routemap1 permit 21
device(config-routemap routemap1)# match ip address 21
device(config)# route-map routemap2 permit 22
```

The route-map permit routes to two networks, and denies the route to one network.

Syntax: [no] route-map *map-name* {permit | deny} *num*

To apply a route map to a RIP interface, enter commands such as the following.

```
device(config-if-e1000-1/1/2)# ip rip route-map map1 in
```

Syntax: [no] ip rip route-map *name* {in | out}

The **route-map** can be a prefix list or an ACL. Setting this command can change the metric.

In applies the route map to routes the Brocade device learns from its neighbor on the interface.

Out applies the route map to routes the Brocade device advertises to its neighbor on the interface.

The commands apply route map map1 as route filters to routes learned from the RIP neighbor on the port.

Setting RIP timers

You can set basic update timers for the RIP protocol. The protocol must be enabled in order to set the timers. The **timers** command specifies how often RIP update messages are sent.

To set the timers, enter the following commands.

```
device(config) router rip
device(config-rip-router)# timer 30 180 180 120
```

Syntax: **[no] timers** *update-timer timeout-timer hold-down-timer garbage-collection-timer*

The *update-timer* parameter sets the amount of time between RIP routing updates. The possible value ranges from 3 - 21845. The default is 30 seconds.

The *timeout-timer* parameter sets the amount of time after which a route is considered unreachable. The possible value ranges from 9 - 65535. The default is 180 seconds.

The *hold-down-timer* parameter sets the amount of time during which information about other paths is ignored. The possible value ranges from 0 - 65535. The default is 180 seconds.

The *garbage-collection-timer* sets the amount of time after which a route is removed from the rip routing table. The possible value ranges from 0 - 65535. The default is 120 seconds.

Displaying RIP Information

To display RIP filters, enter the following command at any CLI level.

```
device# show ip rip
RIP Summary
  Default port 520
    Administrative distance is 120
    Updates every 30 seconds, expire after 180
    Holddown lasts 180 seconds, garbage collect after 120
    Last broadcast 29, Next Update 27
    Need trigger update 0, Next trigger broadcast 1
    Minimum update interval 25, Max update Offset 5
    Split horizon is on; poison reverse is off
    Import metric 1
    Prefix List, Inbound : block_223
    Prefix List, Outbound : block_223
    Route-map, Inbound : Not set
    Route-map, Outbound : Not set
    Redistribute: CONNECTED Metric : 0 Routemap : Not Set
```

No Neighbors are configured in RIP Neighbor Filter Table

Syntax: **show ip rip**

TABLE 41 CLI display of neighbor filter information

Field.	Definition
RIP Summary area	Shows the current configuration of RIP on the device.
Static metric	Shows the static metric configuration. ".not defined" means the route map has not been distributed.
OSPF metric	Shows what OSPF route map has been applied.
Neighbor Filter Table area	
Index	The filter number. You assign this number when you configure the filter.

TABLE 41 CLI display of neighbor filter information (continued)

Field.	Definition
Action	<p>The action the Brocade device takes for RIP route packets to or from the specified neighbor:</p> <ul style="list-style-type: none"> deny - If the filter is applied to an interface's outbound filter group, the filter prevents the Brocade device from advertising RIP routes to the specified neighbor on that interface. If the filter is applied to an interface's inbound filter group, the filter prevents the Brocade device from receiving RIP updates from the specified neighbor. permit - If the filter is applied to an interface's outbound filter group, the filter allows the Brocade device to advertise RIP routes to the specified neighbor on that interface. If the filter is applied to an interface's inbound filter group, the filter allows the Brocade device to receive RIP updates from the specified neighbor.
Neighbor IP Address	The IP address of the RIP neighbor.

To display RIP filters for a specific interface, enter the following command.

```
device# show ip rip interface ethernet 1/1/1
Interface e 1/1/1
RIP Mode : Version2 Running: TRUE
Route summarization disabled
Split horizon is on; poison reverse is off
Default routes not accepted
Metric-offset, Inbound 1
Metric-offset, Outbound 0
Prefix List, Inbound : Not set
Prefix List, Outbound : Not set
Route-map, Inbound : Not set
Route-map, Outbound : Not set
RIP Sent/Receive packet statistics:
Sent : Request 2 Response 34047
Received : Total 123473 Request 1 Response 123472 UnRecognised 0
RIP Error packet statistics:
Rejected 0 Version 0 RespFormat 0 AddrFamily 0
Metric 0 ReqFormat 0
```

Syntax: `show ip rip interface ifName`

To display RIP route information, enter the following command.

```
device# show ip rip route
RIP Routing Table - 474 entries:
1.1.1.1/32, from 169.254.30.1, e 1/1/23 (820)
RIP, metric 4, tag 0, timers: aging 13
1.1.2.1/32, from 169.254.50.1, e 1/3/1 (482)
RIP, metric 3, tag 0, timers: aging 42
1.1.6.1/32, from 169.254.100.1, ve 101 (413)
RIP, metric 2, tag 0, timers: aging 42
169.254.40.0/24, from 192.168.1.2, e 1/1/1 (1894)
RIP, metric 3, tag 0, timers: aging 14
169.254.50.0/24, from 192.168.1.2, e 1/1/1 (1895)
RIP, metric 4, tag 0, timers: aging 14
169.254.100.0/24, from 192.168.1.2, e 1/1/1 (2040)
RIP, metric 2, tag 0, timers: aging 14
169.254.101.0/30, from 192.168.1.2, e 1/1/1 (2105)
223.229.32.0/31, from 169.254.50.1, e 1/3/1 (818)
RIP, metric 2, tag 0, timers: aging 21
```

Syntax: `show ip rip route`

Displaying CPU utilization statistics

To display current running configuration for interface 1/1/1, enter the following command.

```
device# show running-config interface ethernet 1/1/1
interface ethernet 1/1/1
enable
ip ospf area 0
ip ospf priority 0
ip rip v2-only
ip address 10.1.1.2/24
ipv6 address 2000::1/32
ipv6 enable
!
```

To display current running configuration for ve 10, enter the following command.

```
device# show running-config interface ve 10
interface ve 10
ip ospf area 2
ip rip v1-compatible-v2
ip rip poison-reverse
ip address 10.1.0.1/24
ipv6 address 2001:db8:1::14/64
!
```

To display current running configuration for ve 20, enter the following command.

```
device# show running-config interface ve 20
interface ve 20
ip ospf area 1
ip rip v1-only
ip rip poison-reverse
ip address 10.2.0.1/24
!
```

Displaying CPU utilization statistics

You can display CPU utilization statistics for RIP and other IP protocols. To display CPU utilization statistics for RIP, enter the **show cpu-utilization tasks** command at any level of the CLI.

```
device# show cpu-utilization tasks

... Usage average for all tasks in the last 1 second ...
=====
Name                                     %
idle                                    11
con                                      0
mon                                      0
flash                                   0
dbg                                      0
boot                                    0
main                                    0
stkKeepAliveTsk                         0
keygen                                   0
itc                                      0
poeFwdfsm                               0
tmr                                      0
scp                                      0
appl                                    89
snms                                     0
rtm                                      0
rtm6                                    0
rip                                      0
bgp                                      0
bgp_io                                  0
(Output truncated)
```


Syntax: `show cpu-utilization tasks`

The command lists the usage statistics for the previous five-second, one-minute, five-minute, and fifteen-minute intervals.

RIPng

• RIPng Overview.....	219
• Configuring RIPng.....	219
• Clearing RIPng routes from IPv6 route table.....	224
• Displaying RIPng configuration.....	224
• Displaying RIPng routing table.....	225

RIPng Overview

Routing Information Protocol (RIP) is an IP route exchange protocol that uses a distance vector (a number representing a distance) to measure the cost of a given route. RIP uses a hop count as its cost or metric.

IPv6 RIP, known as Routing Information Protocol Next Generation or RIPng, functions similarly to IPv4 RIP version 2. RIPng supports IPv6 addresses and prefixes.

In addition, some new commands that are specific to RIPng have been implemented. This chapter describes the commands that are specific to RIPng. This section does not describe commands that apply to both IPv4 RIP and RIPng.

RIPng maintains a Routing Information Database (RIB), which is a local route table. The local RIB contains the lowest-cost IPv6 routes learned from other RIP routers. In turn, RIPng attempts to add routes from its local RIB into the main IPv6 route table.

Configuring RIPng

To configure RIPng, you must enable RIPng globally on the Brocade device and on individual device interfaces. The following configuration tasks are optional:

- Change the default settings of RIPng timers
- Configure how the Brocade device learns and advertises routes
- Configure which routes are redistributed into RIPng from other sources
- Configure how the Brocade device distributes routes through RIPng
- Configure poison reverse parameters

Enabling RIPng

Before configuring the device to run RIPng, you must do the following:

- Enable the forwarding of IPv6 traffic on the device using the **ipv6 unicast-routing** command.
- Enable IPv6 on each interface over which you plan to enable RIPng. You enable IPv6 on an interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

By default, RIPng is disabled. To enable RIPng, you must enable it globally on the Brocade device and also on individual device interfaces.

NOTE

Enabling RIPng globally on the Brocade device does not enable it on individual device interfaces.

To enable RIPng globally, enter the following command.

```
device(config-rip-router)#ipv6 router rip
device(config-ripng-router)#
```

After you enter this command, the device enters the RIPng configuration level, where you can access several commands that allow you to configure RIPng.

Syntax: [no] ipv6 router rip

To disable RIPng globally, use the **no** form of this command.

After enabling RIPng globally, you must enable it on individual Brocade device interfaces. You can enable it on physical as well as virtual routing interfaces. For example, to enable RIPng on Ethernet interface 1/3/1, enter the following commands.

```
device(config)# interface ethernet 1/3/1
device(config-if-e100-1/3/1)# ipv6 rip enable
```

Syntax: [no] ipv6 rip enable

To disable RIPng on an individual device interface, use the **no** form of this command.

Configuring RIPng timers

TABLE 42 RIPng timers

Timer	Description	Default
Update	Amount of time (in seconds) between RIPng routing updates.	30 seconds.
Timeout	Amount of time (in seconds) after which a route is considered unreachable.	180 seconds.
Hold-down	Amount of time (in seconds) during which information about other paths is ignored.	180 seconds.
Garbage-collection	Amount of time (in seconds) after which a route is removed from the routing table.	120 seconds.

You can adjust these timers for RIPng. Before doing so, keep the following caveats in mind:

- If you adjust these RIPng timers, Brocade strongly recommends setting the same timer values for all routers and access servers in the network.
- Setting the update timer to a shorter interval can cause the devices to spend excessive time updating the IPv6 route table.
- Brocade recommends setting the timeout timer value to at least three times the value of the update timer.
- Brocade recommends a shorter hold-down timer interval, because a longer interval can cause delays in RIPng convergence.

The following example sets updates to be advertised every 45 seconds. If a route is not heard from in 135 seconds, the route is declared unusable. Further information is suppressed for an additional 10 seconds. Assuming no updates, the route is flushed from the routing table 20 seconds after the end of the hold-down period.

```
device(config)# ipv6 router rip
device(config-ripng-router)# timers 45 135 10 20
```

Syntax: [no] timersupdate-timer timeout-timer hold-down-timer garbage-collection-timer

Possible values for the timers are as follows:

- Update timer: 3 through 65535 seconds.
- Timeout timer: 9 through 65535 seconds.

- Hold-down timer: 9 through 65535 seconds.
- Garbage-collection timer: 9 through 65535 seconds.

NOTE

You must enter a value for each timer, even if you want to retain the current setting of a particular timer.

To return to the default values of the RIPng timers, use the **no** form of this command.

Configuring route learning and advertising parameters

You can configure the following learning and advertising parameters:

- Learning and advertising of RIPng default routes.
- Advertising of IPv6 address summaries.
- Metric of routes learned and advertised on a Brocade device interface.

Configuring default route learning and advertising

By default, the device does not learn IPv6 default routes (::/0). You can originate default routes into RIPng, which causes individual Brocade device interfaces to include the default routes in their updates. When configuring the origination of the default routes, you can also do the following:

- Suppress all other routes from the updates.
- Include all other routes in the updates.

For example, to originate default routes in RIPng and suppress all other routes in updates sent from Ethernet interface 1/3/1, enter the following commands.

```
device(config)# interface ethernet 1/3/1
device(config-if-e100-1/3/1)# ipv6 rip default-information only
```

To originate IPv6 default routes and include all other routes in updates sent from Ethernet interface 1/3/1, enter the following commands.

```
device(config)# interface ethernet 1/3/1
device(config-if-e100-1/3/1)# ipv6 rip default-information originate
```

Syntax: **[no] ipv6 rip default-information { only | originate }**

The **only** keyword originates the default routes and suppresses all other routes from the updates.

The **originate** keyword originates the default routes and includes all other routes in the updates.

To remove the explicit default routes from RIPng and suppress advertisement of these routes, use the **no** form of this command.

Advertising IPv6 address summaries

You can configure RIPng to advertise a summary of IPv6 addresses from a Brocade device interface and to specify an IPv6 prefix that summarizes the routes.

If a route's prefix length matches the value specified in the **ipv6 rip summary-address** command, RIPng advertises the prefix specified in the **ipv6 rip summary-address** command instead of the original route.

For example, to advertise the summarized prefix 2001:db8::/36 instead of the IPv6 address 2001:db8:0:adff:8935:e838:78:e0ff with a prefix length of 64 bits from Ethernet interface 1/3/1, enter the following commands.

```
device(config)# interface ethernet 1/3/1
device(config-if-e100-1/3/1)# ipv6 address 2001:db8:0:adff:8935:e838:78:
e0ff /64
device(config-if-e100-1/3/1)# ipv6 rip summary-address 2001:db8::/36
```

Syntax: [no] `ipv6 rip summary-address ipv6-prefix/prefix-length`

You must specify the *ipv6-prefix* parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

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.

To stop the advertising of the summarized IPv6 prefix, use the **no** form of this command.

Changing the metric of routes learned and advertised on an interface

A device interface increases the metric of an incoming RIPng route it learns by an offset (the default is one). The device then places the route in the route table. When the device sends an update, it advertises the route with the metric plus the default offset of zero in an outgoing update message.

You can change the metric offset an individual interface adds to a route learned by the interface or advertised by the interface. For example, to change the metric offset for incoming routes learned by Ethernet interface 1/3/1 to one and the metric offset for outgoing routes advertised by the interface to three, enter the following commands.

```
device(config)# interface ethernet 1/3/1
device(config-if-e100-1/3/1)# ipv6 rip metric-offset 2
device(config-if-e100-1/3/1)# ipv6 rip metric-offset out 3
```

In this example, if Ethernet interface 1/3/1 learns about an incoming route, it will increase the incoming metric by two. If the interface 1/3/1 advertises an outgoing route, it will increase the metric offset by 3 as specified in the example. Configuring the default metric (1 for incoming, 0 for outgoing) will be allowed but will not be visible in the **show run** output for the interface.

Syntax: [no] `ipv6 rip metric-offset 1-16`

Syntax: [no] `ipv6 rip metric-offset out 0-15`

To return the metric offset to its default value, use the **no** form of this command.

Redistributing routes into RIPng

You can configure the Brocade device to redistribute routes from the following sources into RIPng:

- IPv6 static routes
- Directly connected IPv6 networks
- BGP4+
- OSPFv3

When you redistribute a route from BGP4+ or OSPFv3 into RIPng, the device can use RIPng to advertise the route to its RIPng neighbors.

When configuring the Brocade device to redistribute routes, such as BGP4+ routes, you can optionally specify a metric for the redistributed routes. If you do not explicitly configure a metric, the default metric value of one is used.

For example, to redistribute OSPFv3 routes into RIPng, enter the following command.

```
device(config)# ipv6 router rip
device(config-ripng-router)# redistribute ospf
```

Syntax: `[no] redistribute{ bgp | connected | ospf | static [metric number] }`

For the metric, specify a numerical value that is consistent with RIPng.

Controlling distribution of routes through RIPng

You can create a prefix list and then apply it to RIPng routing updates that are received or sent on a device interface. Performing this task allows you to control the distribution of routes through RIPng.

For example, to permit the inclusion of routes with the prefix 2001:db8::/32 in RIPng routing updates sent from Ethernet interface 1/3/1, enter the following commands.

```
device(config)# ipv6 prefix-list routesfor2001 permit 2001:db8::/32
device(config)# ipv6 router rip
device(config-ripng-router)# distribute-list prefix-list routesfor2001 out
```

To deny prefix lengths greater than 64 bits in routes that have the prefix 2001:db8::/64 and allow all other routes received on tunnel interface 1/3/1, enter the following commands.

```
device(config)# ipv6 prefix-list 2001routes deny 2001:db8::/64 le 128
device(config)# ipv6 prefix-list 2001routes permit ::/0 ge 0 le 128
device(config)# ipv6 router rip
device(config-ripng-router)# distribute-list prefix-list 2001routes in
```

Syntax: `[no] distribute-list prefix-list name { in | out }`

The *name* parameter indicates the name of the prefix list generated using the **ipv6 prefix-list** command.

The **in** keyword indicates that the prefix list is applied to incoming routing updates on the specified interface.

The **out** keyword indicates that the prefix list is applied to outgoing routing updates on the specified interface.

To remove the distribution list, use the **no** form of this command.

Configuring poison reverse parameters

By default, poison reverse is disabled on a RIPng Brocade device. If poison reverse is enabled, RIPng advertises routes it learns from a particular interface over that same interface with a metric of 16, which means that the route is unreachable.

Enabling poison reverse on the RIPng Brocade device disables split-horizon and vice versa. By default, split horizon will be enabled.

To enable poison reverse on the RIPng Brocade device, enter the following commands.

```
device(config)# ipv6 router rip
device(config-ripng-router)# poison-reverse
```

Syntax: `[no] poison-reverse`

To disable poison-reverse, use the **no** form of this command.

By default, if a RIPng interface goes down, the Brocade device does not send a triggered update for the interface's IPv6 networks.

Clearing RIPng routes from IPv6 route table

To better handle this situation, you can configure a RIPng Brocade device to send a triggered update containing the local routes of the disabled interface with an unreachable metric of 16 to the other RIPng routers in the routing domain. You can enable the sending of a triggered update by entering the following commands.

```
device(config)# ipv6 router rip
device(config-ripng-router)# poison-local-routes
```

Syntax: `[no] poison-local-routes`

To disable the sending of a triggered update, use the **no** form of this command.

Clearing RIPng routes from IPv6 route table

To clear all RIPng routes from the RIPng route table and the IPv6 main route table and reset the routes, enter the following command at the Privileged EXEC level or any of the configuration levels of the CLI.

```
device# clear ipv6 rip route
```

Syntax: `clear ipv6 rip route`

Displaying RIPng configuration

To display RIPng configuration information, enter the **show ipv6 rip** command at any CLI level.

```
device# show ipv6 rip
IPv6 rip enabled, port 521
  Administrative distance is 120
  Updates every 30 seconds, expire after 180
  Holddown lasts 180 seconds, garbage collect after 120
  Split horizon is on; poison reverse is off
  Default routes are not generated
  Periodic updates 5022, trigger updates 10
  Distribute List, Inbound : Not set
  Distribute List, Outbound : Not set
  Redistribute: CONNECTED
```

Syntax: `show ipv6 rip`

TABLE 43 show ipv6 rip output descriptions

Field	Description
IPv6 RIP status/port	The status of RIPng on the device. Possible status is "enabled" or "disabled." The UDP port number over which RIPng is enabled.
Administrative distance	The setting of the administrative distance for RIPng.
Updates/expiration	The settings of the RIPng update and timeout timers.
Holddown/garbage collection	The settings of the RIPng hold-down and garbage-collection timers.
Split horizon/poison reverse	The status of the RIPng split horizon and poison reverse features. Possible status is "on" or "off."
Default routes	The status of RIPng default routes.
Periodic updates/trigger updates	The number of periodic updates and triggered updates sent by the RIPng Brocade device.
Distribution lists	The inbound and outbound distribution lists applied to RIPng.

TABLE 43 show ipv6 rip output descriptions (continued)

Field	Description
Redistribution	<p>The types of IPv6 routes redistributed into RIPng. The types can include the following:</p> <ul style="list-style-type: none"> • STATIC - IPv6 static routes are redistributed into RIPng. • CONNECTED - Directly connected IPv6 networks are redistributed into RIPng. • BGP - BGP4+ routes are redistributed into RIPng. • OSPF - OSPFv3 routes are redistributed into RIPng.

Displaying RIPng routing table

To display the RIPng routing table, enter the following command at any CLI level.

```
device# show ipv6 rip route
IPv6 RIP Routing Table - 4 entries:
ada::1:1:1:2/128, from fe80::224:38ff:fe8f:3000, e 1/3/4
    RIP, metric 2, tag 0, timers: aging 17
2001:db8::/64, from fe80::224:38ff:fe8f:3000, e 1/3/4
    RIP, metric 3, tag 0, timers: aging 17
bebe::1:1:1:4/128, from ::, null (0)
    CONNECTED, metric 1, tag 0, timers: none
cccc::1:1:1:3/128, from fe80::768e:f8ff:fe94:2da, e 2/1/23
    RIP, metric 2, tag 0, timers: aging 50
```

Syntax: `show ipv6 rip route [ipv6-prefix/prefix-length | ipv6-address]`

The *ipv6-prefix/prefix-length* parameters restrict the display to the entries for the specified IPv6 prefix. You must specify the *ipv6-prefix* parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. 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.

The *ipv6-address* parameter restricts the display to the entries for the specified IPv6 address. You must specify this parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

TABLE 44 show ipv6 rip route output descriptions

Field	Description
IPv6 RIP Routing Table entries	The total number of entries in the RIPng routing table.
<i>ipv6-prefix /prefix-length</i>	The IPv6 prefix and prefix length.
<i>ipv6-address</i>	The IPv6 address.
Next-hop router	The next-hop router for this Brocade device. If :: appears, the route is originated locally.
Interface	The interface name. If "null" appears, the interface is originated locally.
Source of route	<p>The source of the route information. The source can be one of the following:</p> <ul style="list-style-type: none"> • RIP - routes learned by RIPng. • CONNECTED - IPv6 routes redistributed from directly connected networks. • STATIC - IPv6 static routes are redistributed into RIPng. • BGP - BGP4+ routes are redistributed into RIPng. • OSPF - OSPFv3 routes are redistributed into RIPng.
Metric <i>number</i>	The cost of the route. The <i>number</i> parameter indicates the number of hops to the destination.

Displaying RIPv6 routing table

TABLE 44 show ipv6 rip route output descriptions (continued)

Field	Description
Tag <i>number</i>	The tag value of the route.
Timers	Indicates if the hold-down timer or the garbage-collection timer is set.

OSPFv2

• OSPFv2 overview.....	228
• Autonomous System.....	228
• OSPFv2 components and roles.....	229
• Reduction of equivalent AS external LSAs.....	230
• Algorithm for AS external LSA reduction.....	232
• Enabling OSPFv2.....	232
• Backbone area.....	232
• Assigning OSPFv2 areas.....	233
• Area range.....	233
• Area types.....	234
• Stub area and totally stubby area.....	234
• Not-so-stubby area (NSSA).....	235
• Assigning interfaces to an area.....	237
• Link state advertisements.....	238
• Virtual links.....	238
• Default route origination.....	240
• External route summarization.....	241
• SPF timers.....	241
• Modifying Shortest Path First timers.....	242
• OSPFv2 LSA refreshes.....	242
• Support for OSPF RFC 2328 Appendix E.....	243
• OSPFv2 graceful restart.....	244
• OSPFv2 stub router advertisement.....	246
• OSPFv2 Shortest Path First throttling.....	246
• IETF RFC and internet draft support.....	247
• OSPFv2 non-stop routing.....	247
• Synchronization of critical OSPFv2 elements.....	248
• Standby module operations.....	249
• OSPFv2 distribute list.....	250
• OSPFv2 route redistribution.....	252
• Redistributing routes into OSPFv2.....	253
• Load sharing.....	254
• Interface types to which the reference bandwidth does not apply.....	255
• Changing the reference bandwidth for the cost on OSPFv2 interfaces.....	255
• OSPFv2 over VRF.....	256
• Configuring the OSPFv2 Max-Metric Router LSA.....	257
• Re-enabling OSPFv2 compatibility with RFC 1583.....	257
• Changing default settings.....	257
• Disabling and re-enabling OSPFv2 event logging.....	258
• Understanding the effects of disabling OSPFv2.....	258
• Disabling OSPFv2.....	258
• Displaying OSPFv2 results.....	259

OSPFv2 overview

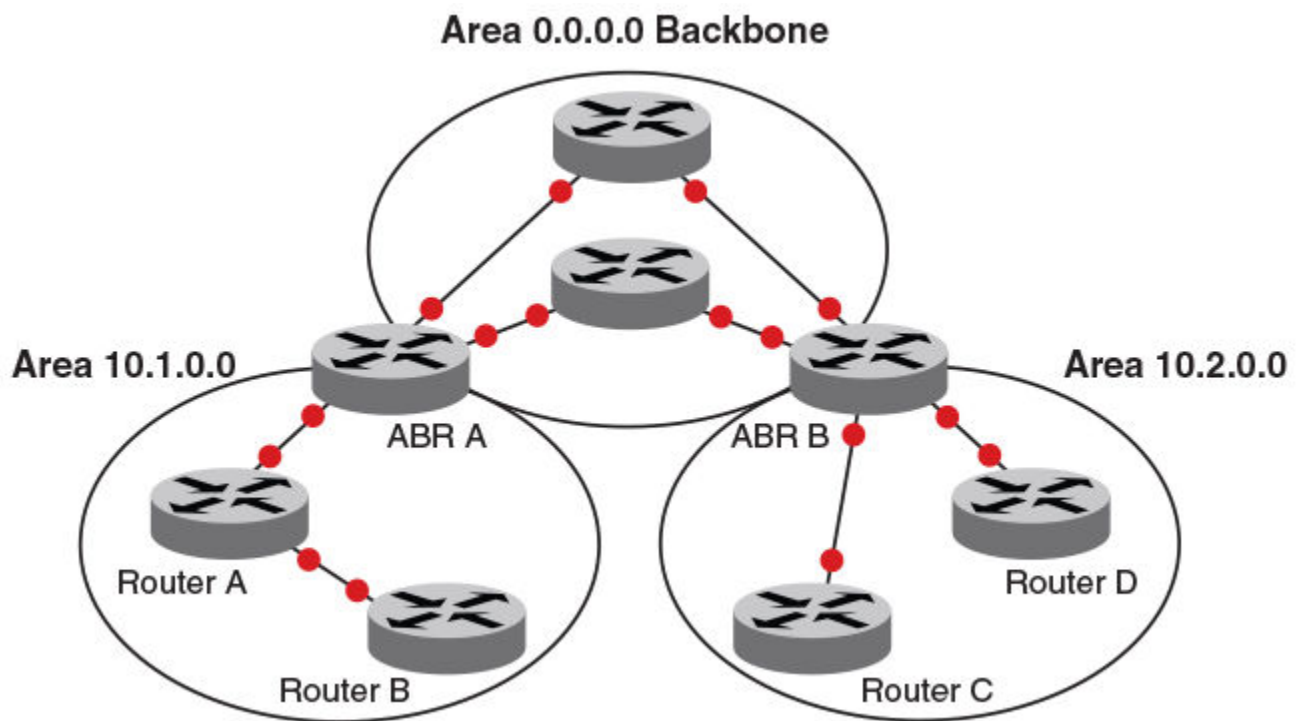
Open Shortest Path First Version 2 (OSPFv2) is a link-state routing protocol that uses link-state advertisements (LSAs) to update neighboring routers about a router's interfaces. Each router maintains an identical area-topology database to determine the shortest path to any neighboring router.

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 21 OSPF operating in a network



NOTE

For details of components and virtual links, refer to [OSPFv2 components and roles](#) on page 229 and [Virtual links](#) on page 238, 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.

For more information about redistribution, refer to the **redistribute** command in the *FastIron Command Reference*.

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 DR and other (a router that is not a DR or a BDR) and its DR and BDR.

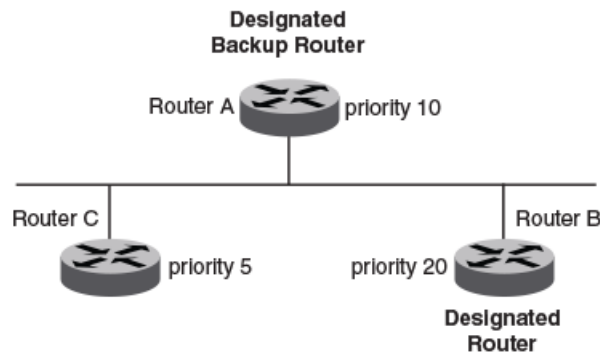
NOTE

In an OSPF point-to-point network, where a direct Layer 3 connection exists between a single pair of OSPF routers, there is no need for designated or backup designated routers.

Without the need for Designated and Backup Designated routers, a point-to-point network establishes adjacency and converges faster. The neighboring routers become adjacent whenever they can communicate directly. In contrast, in broadcast and non-broadcast multi-access (NBMA) networks, the Designated Router and Backup Designated Router become adjacent to all other routers attached to the network.

In a network with no designated router and no backup designated router, the neighboring router with the highest priority is elected as the DR, and the router with the next highest priority is elected as the BDR, as shown in the figure below. Priority is a configurable option at the interface level; refer to the **ip ospf priority** command in the *FastIron Command Reference*.

FIGURE 22 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 Brocade device's router ID is the IP address configured on the lowest numbered loopback interface. If the device does not have a loopback interface, the default router ID is the lowest numbered IP address configured on the device.

When multiple routers on the same network are declaring themselves DRs, then both the priority and router ID are used to select the designated router and backup designated routers.

The DR and BDR election process is performed when one of the following events occurs:

- An interface is in a waiting state and the wait time expires.
- An interface is in a waiting state and receives a hello packet that addresses the BDR.
- A change in the neighbor state occurs, such as the following:
 - A neighbor state transitions from ATTEMPT state to a higher state.
 - Communication to a neighbor is lost.
 - A neighbor declares itself to be the DR or BDR for the first time.

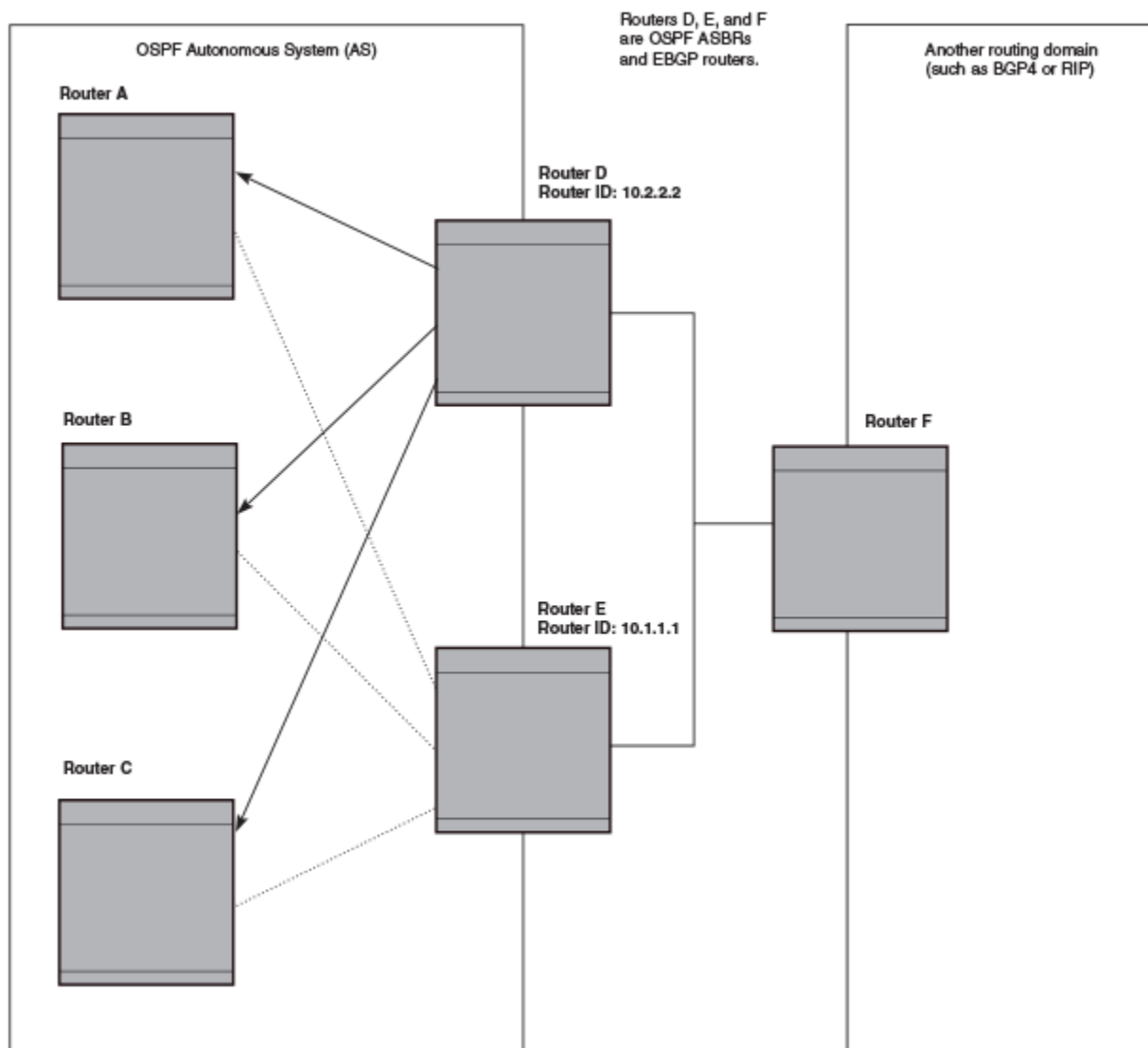
Reduction of equivalent AS external LSAs

An OSPF ASBR uses AS External link advertisements (AS External LSAs) to originate advertisements of a route learned from another routing domain, such as a BGP4 or RIP domain. The ASBR advertises the route to the external domain by flooding AS External LSAs to all the other OSPF devices (except those inside stub networks) within the local OSPF Autonomous System (AS).

In some cases, multiple ASBRs in an AS can originate equivalent LSAs. The LSAs are equivalent when they have the same cost, the same next hop, and the same destination. The device optimizes OSPF by eliminating duplicate AS External LSAs in this case. The device with the lower router ID flushes the duplicate External LSAs from its database and thus does not flood the duplicate External LSAs into the OSPF AS. AS External LSA reduction, therefore, reduces the size of the link state database on the device. The AS External LSA reduction is described in RFC 2328

In this example, Routers D and E are OSPF ASBRs, and thus communicate route information between the OSPF AS, which contains Routers A, B, and C, and another routing domain, which contains Router F. The other routing domain is running another routing protocol, such as BGP4 or RIP. Routers D, E, and F, therefore, are each running both OSPF and either BGP4 or RIP.

FIGURE 23 AS external LSA reduction



Notice that both Router D and Router E have a route to the other routing domain through Router F.

OSPF eliminates the duplicate AS External LSAs. When two or more devices are configured as ASBRs have equal-cost routes to the same next-hop router in an external routing domain, the ASBR with the highest router ID floods the AS External LSAs for the external domain into the OSPF AS, while the other ASBRs flush the equivalent AS External LSAs from their databases. As a result, the overall volume of route advertisement traffic within the AS is reduced and the devices that flush the duplicate AS External LSAs have more memory for other OSPF data. Because Router D has a higher router ID than Router E, Router D floods the AS External LSAs for Router F to Routers A, B, and C. Router E flushes the equivalent AS External LSAs from its database.

Algorithm for AS external LSA reduction

The AS external LSA reduction example shows the normal AS External LSA reduction feature. The behavior changes under the following conditions:

- There is one ASBR advertising (originating) a route to the external destination, but one of the following happens:
 - A second ASBR comes on-line
 - A second ASBR that is already on-line begins advertising an equivalent route to the same destination.

In either case above, the router with the higher router ID floods the AS External LSAs and the other router flushes its equivalent AS External LSAs. For example, if Router D is offline, Router E is the only source for a route to the external routing domain. When Router D comes on-line, it takes over flooding of the AS External LSAs to Router F, while Router E flushes its equivalent AS External LSAs to Router F.

- One of the ASBRs starts advertising a route that is no longer equivalent to the route the other ASBR is advertising. In this case, the ASBRs each flood AS External LSAs. Since the LSAs either no longer have the same cost or no longer have the same next-hop router, the LSAs are no longer equivalent, and the LSA reduction feature no longer applies.
- The ASBR with the higher router ID becomes unavailable or is reconfigured so that it is no longer an ASBR. In this case, the other ASBR floods the AS External LSAs. For example, if Router D goes off-line, then Router E starts flooding the AS with AS External LSAs for the route to Router F.

Enabling OSPFv2

A number of steps are required when enabling OSPFv2 on a device.

Consider the following when enabling OSPFv2 on a device.

- Redistribution must be enabled on devices configured to operate as ASBRs.
 - All device ports must be assigned to one of the defined areas on an OSPF device. When a port is assigned to an area, all corresponding subnets on that port are automatically included in the assignment.
1. Enter the **router ospf** command in global configuration mode to enable OSPF on the device.
 2. 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 257.

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-ospf-router)# area 0
```

4. Enter the **area** command to define a second OSPFv2 area ID.

```
device(config-ospf-router)# area 10.1.1.1
```

The following example assigns an OSPFv2 ID to two areas. One of the areas is assigned by decimal number. The second area is assigned by IP address.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# area 0
device(config-ospf-router)# area 10.1.1.1
```

Area range

You can further consolidate routes at an area boundary by defining an area range. The area range allows you to assign an aggregate address to a range of IP and IPv6 addresses.

This aggregate value becomes the address that is advertised instead of all the individual addresses it represents being advertised. Only this aggregate or summary address is advertised into other areas instead of all the individual addresses that fall in the configured range. Area range configuration can considerably reduce the number of Type 3 summary LSAs advertised by a device. You have the option of adding the cost to the summarized route. If you do not specify a value, the cost value is the default range metric calculation for the generated summary LSA cost. You can temporarily pause route summarization from the area by suppressing the type 3 LSA so that the component networks remain hidden from other networks.

You can assign up to 32 ranges in an OSPF area.

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. Repeat as necessary.

```
device(config-ospf-router)# area 10.0.0.10 range 10.45.0.0 10.255.0.0
device(config-ospf-router)# area 10.0.0.20 range 10.45.0.0 10.255.0.0
```

The following example defines an area range for subnets on 10.0.0.10 and 10.0.0.20.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# area 10.0.0.10 range 10.45.0.0 10.255.0.0
device(config-ospf-router)# area 10.0.0.20 range 10.45.0.0 10.255.0.0
```

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-ospf-router)# area 40 stub 99 no-summary
```

The following example configures a stub area, specifying a cost of 99 and preventing any Type 3 and Type 4 summary LSAs from being injected into the area.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# area 40 stub 99 no-summary
```

Not-so-stubby area (NSSA)

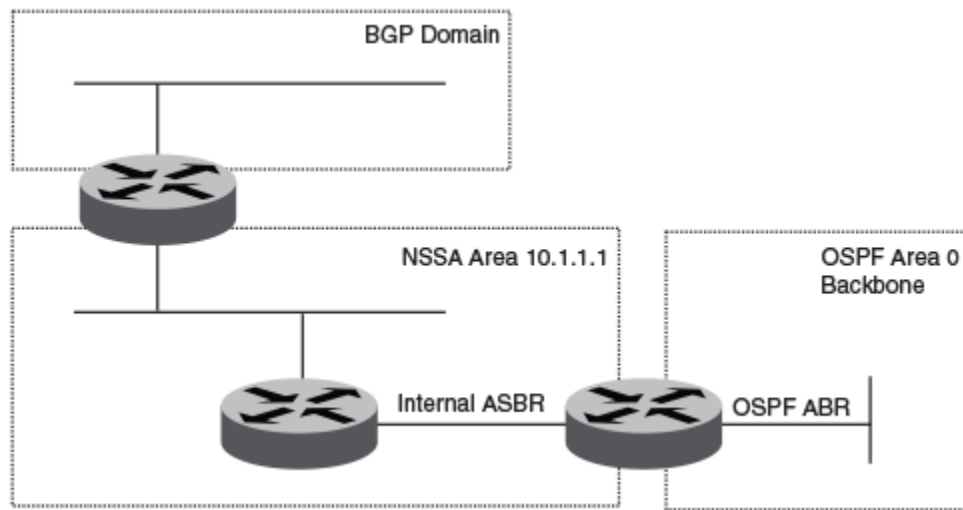
The OSPFv2 not-so-stubby area (NSSA) enables you to configure OSPFv2 areas that provide the benefits of stub areas, but that also are capable of importing external route information. OSPFv2 does not flood external routes from other areas into an NSSA, but does translate and flood route information from 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.

Not-so-stubby area (NSSA)

FIGURE 24 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-ospf-router)# area 10.1.1.1 nssa 1
```

Area 10.1.1.1 is defined as an NSSA.

The following example configures OSPF area 10.1.1.1 as an NSSA.

```
device# configure terminal
device(router ospf
device(config-ospf-router)# area 10.1.1.1 nssa 1
```

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-ospf-router)# 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-ospf-router)# summary-address 10.10.1.0 10.10.2.0
```

The following example configures a summary-address in NSSA 10.1.1.1.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# area 10.1.1.1 nssa 10
device(config-ospf-router)# summary-address 10.10.1.0 10.10.2.0
```

Assigning interfaces to an area

Once you define OSPFv2 areas, you can assign interfaces to the areas. All device ports must be assigned to one of the defined areas on an OSPFv2 device. When a port is assigned to an area, all corresponding subnets on that port are automatically included in the assignment.

To assign a loopback interface to an area with the IP address of 10.5.0.0, perform the following task:

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

```
device# configure terminal
```

2. Enter the **interface** command and specify an interface.

```
device(config)# interface loopback 2
```

3. Enter the **ip ospf area** command followed by the IP address of the area.

```
device(config-lbif-2)# ip ospf area 10.5.0.0
```

If you want to set an interface to passive mode, use the **ip ospf passive** command. If you want to block flooding of outbound LSAs on specific OSPF interfaces, use the **ip ospf database-filter all out** command. (Refer to the *Brocade FastIron Command Reference* for details.)

The following example assigns a loopback interface to an area with the IP address of 10.5.0.0.

```
device# configure terminal
device(config)# interface loopback 2
device(config-lbif-2)# ip ospf area 10.5.0.0
```

Link state advertisements

Brocade devices support the following types of LSAs, which are described in RFC 2328 and 3101:

- Router link
- Network link
- Summary link
- Autonomous system summary link
- AS external link
- Not-So-Stubby Area (NSSA) external link
- Grace LSAs

Communication among areas is provided by means of link state advertisements (LSAs). The LSAs supported for each area type are as follows:

- Backbone (area 0) supports LSAs 1, 2, 3, 4, 5, and 7.
- Nonbackbone area supports LSAs 1, 2, 3, 4, and 5.
- Stub area supports LSAs 1, 2, and 3.
- Totally stubby area (TSA) supports LSAs 1 and 2, and also supports a single LSA 3 per ABR, advertising a default route.
- No so stubby area (NSSA) supports LSAs 1, 2, 3, and 7.

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.

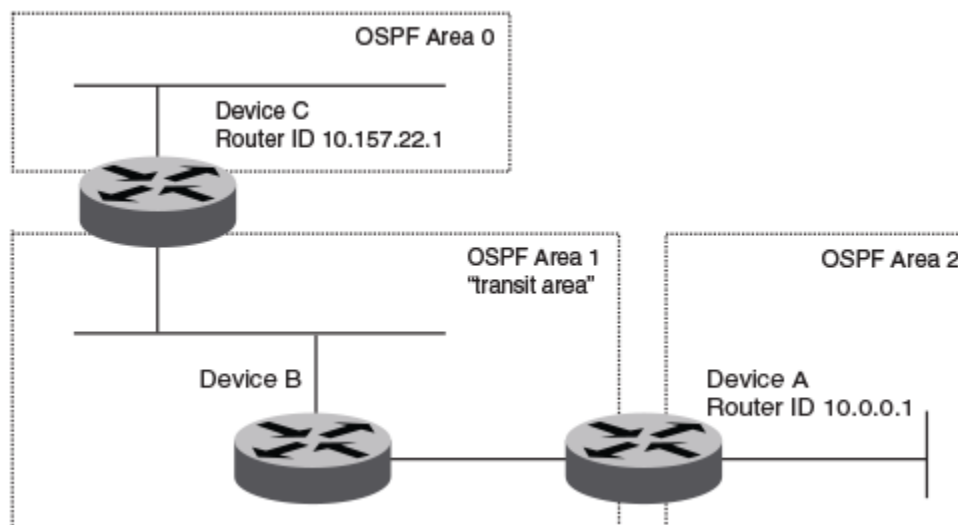
NOTE

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 25 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-ospf-router)# area 0
```

4. Enter the **area** command to assign an OSPFv2 area ID.

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

Default route origination

8. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-ospf-router)# area 1
```

9. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-ospf-router)# 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-ospf-router)# area 1 virtual-link 10.1.1.1
```

The following example configures a virtual link between two devices.

```
ABR1:
device1# configure terminal
device1(config)# router ospf
device1(config-ospf-router)# area 0
device1(config-ospf-router)# area 1
device1(config-ospf-router)# area 1 virtual-link 10.2.2.2

ABR2:
device2# configure terminal
device2(config)# router ospf
device2(config-ospf-router)# area 1
device2(config-ospf-router)# area 2
device2(config-ospf-router)# area 1 virtual-link 10.1.1.1
```

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 iBGP neighbor when default-information-originate is configured. The device does not, however, originate the default route if the active default route is learned from an OSPFv2 device in the same domain.

NOTE

The device does not advertise the OSPFv2 default route, regardless of other configuration parameters, unless you explicitly enable default route origination.

If default route origination is enabled and you disable it, the default route originated by the device is flushed. Default routes generated by other OSPFv2 devices are not affected. If you re-enable the default route origination, the change takes effect immediately and you do not need to reload the software.

External route summarization

An ASBR can be configured to advertise one external route as an aggregate for all redistributed routes that are covered by a specified address range.

When you configure a summary address range, the range takes effect immediately. All the imported routes are summarized according to the configured summary address range. Imported routes that have already been advertised and that fall within the range are flushed out of the autonomous system and a single route corresponding to the range is advertised.

If a route that falls within a configured summary address range is imported by the device, no action is taken if the device has already advertised the aggregate route; otherwise, the device advertises the aggregate route. If an imported route that falls within a configured summary address range is removed by the device, no action is taken if there are other imported routes that fall within the same summary address range; otherwise, the aggregate route is flushed.

You can configure up to 32 summary address ranges. The device sets the forwarding address of the aggregate route to 0 and sets the tag to 0. If you delete a summary address range, the advertised aggregate route is flushed and all imported routes that fall within the range are advertised individually. If an external link-state database (LSDB) overflow condition occurs, all aggregate routes and other external routes are flushed out of the autonomous system. When the device exits the external LSDB overflow condition, all the imported routes are summarized according to the configured summary address ranges.

NOTE

If you use redistribution filters in addition to summary address ranges, the device applies the redistribution filters to routes first, and then applies them to the summary address ranges.

NOTE

If you disable redistribution, all the aggregate routes are flushed, along with other imported routes.

NOTE

This option affects only imported, type 5 external LSA routes. A single type 5 LSA is generated and flooded throughout the autonomous system for multiple external routes. Type 7-route redistribution is not affected by this feature. All type 7 routes will be imported (if redistribution is enabled). To summarize type 7 LSAs or exported routes, use NSSA address range summarization.

SPF timers

The device uses an SPF delay timer and an SPF hold-time timer to calculate the shortest path for OSPFv2 routes. The values for both timers can be changed.

- **SPF delay:** When the device receives a topology change, it waits before starting a Shortest Path First (SPF) calculation. By default, the device waits zero seconds. You can configure the SPF delay to a value from 0 through 65535 seconds. If you set the SPF delay to 0 seconds, the device immediately begins the SPF calculation after receiving a topology change.
- **SPF hold time:** The device waits a specific amount of time between consecutive SPF calculations. By default, it waits zero seconds. You can configure the SPF hold time to a value from 0 through 65535 seconds. If you set the SPF hold time to 0 seconds, the device does not wait between consecutive SPF calculations.

You can set the SPF delay and hold time to lower values to cause the device to change to alternate paths more quickly if a route fails. Note that lower values for these parameters require more CPU processing time.

You can change one or both of the timers.

NOTE

If you want to change only one of the timers, for example, the SPF delay timer, you must specify the new value for this timer as well as the current value of the SPF hold timer, which you want to retain. The device does not accept only one timer value.

NOTE

If you configure SPF timers between 0 through 100, they default to 0.

Modifying Shortest Path First timers

The Shortest Path First (SPF) throttle timers can be modified.

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

```
device# configure terminal
```

2. Enter the **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-ospf-router)# timers throttle spf 100 500 5000
```

The following example sets the SPF initial delay to 100 milliseconds, the hold time to 500 milliseconds, and the maximum wait time to 5000 milliseconds.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# timers throttle spf 100 500 5000
```

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-ospf-router)# timers lsa-group-pacing 120
```

The OSPFv2 LSA pacing interval is changed to 120 seconds (2 minutes).

The following example changes the OSPFv2 LSA pacing interval is changed to 120 seconds (2 minutes).

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# timers lsa-group-pacing 120
```

Support for OSPF RFC 2328 Appendix E

Brocade devices support Appendix E in OSPF RFC 2328. Appendix E describes a method to ensure that an OSPF device generates unique link state IDs for type-5 (External) link state advertisements (LSAs) in cases where two networks have the same network address but different network masks.

NOTE

Support for Appendix E of RFC 2328 is enabled automatically and cannot be disabled. No user configuration is required.

Normally, an OSPF device uses the network address alone for the link state ID of the link state advertisement (LSA) for the network. For example, if the device needs to generate an LSA for network 10.1.2.3 255.0.0.0, the device generates ID 10.1.2.3 for the LSA.

However, suppose that an OSPF device needs to generate LSAs for all the following networks:

- 10.0.0.0 255.0.0.0
- 10.0.0.0 255.255.0.0
- 10.0.0.0 255.255.255.0

All three networks have the same network address, 10.0.0.0. Without support for RFC 2328 Appendix E, an OSPF device uses the same link state ID, 10.0.0.0, for the LSAs for all three networks. For example, if the device generates an LSA with ID 10.0.0.0 for network 10.0.0.0 255.0.0.0, this LSA conflicts with the LSA generated for network 10.0.0.0 255.255.0.0 or 10.0.0.0 255.255.255.0. The result is multiple LSAs that have the same ID but that contain different route information.

When appendix E is supported, the device generates the link state ID for a network as the following steps.

1. Does an LSA with the network address as its ID already exist?
 - No - Use the network address as the ID.
 - Yes - Go to "Support for OSPF RFC 2328 Appendix E".

2. Compare the networks that have the same network address, to determine which network is more specific. The more specific network is the one that has more contiguous one bits in its network mask. For example, network 10.0.0.0 255.255.0.0 is more specific than network 10.0.0.0 255.0.0.0, because the first network has 16 ones bits (255.255.0.0) whereas the second network has only 8 ones bits (255.0.0.0).
 - – For the less specific network, use the network's address as the ID.
 - For the more specific network, use the network's broadcast address as the ID. The broadcast address is the network address, with all ones bits in the host portion of the address. For example, the broadcast address for network 10.0.0.0 255.255.0.0 is 10.0.255.255.

If this comparison results in a change to the ID of an LSA that has already been generated, the device generates a new LSA to replace the previous one. For example, if the device has already generated an LSA for network with ID 10.0.0.0 for network 10.0.0.0 255.255.255.0, the device must generate a new LSA for the network, if the device needs to generate an LSA for network 10.0.0.0 255.255.0.0 or 10.0.0.0 255.0.0.0.

OSPFv2 graceful restart

The graceful restart (GR) feature provides a routing device with the capability to inform its neighbors when it is performing a restart.

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.

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-ospf-router)# no graceful-restart
```

The following example disables GR.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# 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-ospf-router)# 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-ospf-router)# graceful-restart restart-time 240
```

The following example re-enables GR and changes the maximum restart wait time from the default value of 120 seconds to 240 seconds.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# graceful-restart
device(config-ospf-router)# graceful-restart restart-time 240
```

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-ospf-router)# graceful-restart helper-disable
```

The following example disables the GR helper.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# graceful-restart helper-disable
```

OSPFv2 stub router advertisement

OSPFv2 stub router advertisement is an open standard based feature and it is specified in RFC 3137. This feature provides a user with the ability to gracefully introduce and remove an OSPFv2 device from the network, by controlling when the data traffic can start and stop flowing through the device in cases where there are other OSPFv2 devices present on the network providing alternative paths for the traffic. This feature does not work if there is no alternative for the traffic through other OSPFv2 routers. The device can control the data traffic flowing through it by changing the cost of the paths passing through the configured device. By setting the path cost high the traffic will be redirected to other OSPFv2 devices providing a lower cost path. This change in path cost is accomplished by setting the metric of the links advertised in the Router LSA to a maximum value. When the OSPFv2 device is ready to forward the traffic, the links are advertised with the real metric value instead of the maximum value.

OSPFv2 stub router advertisement is useful for avoiding a loss of traffic during short periods when adjacency failures are detected and traffic is rerouted. Using this feature, traffic can be rerouted before an adjacency failure occurs due to common services interruptions such as a router being shutdown for maintenance.

OSPFv2 stub router advertisement is also useful during startup because it gives the device enough time to build up its routing table before forwarding traffic. This can be useful where BGP is enabled on the device because it takes time for the BGP routing table to converge.

You can also configure and set a metric value for the following LSA types:

- Summary (type 3 and type 4)
- External (type 5 and type 7)
- Opaque (type 10, TE link)

OSPFv2 Shortest Path First throttling

Rapid triggering of SPF calculations with exponential back-off to offer the advantages of rapid convergence without sacrificing stability. As the delay increases, multiple topology changes can occur within a single SPF. This dampens network activity due to frequent topology changes.

This scheduling method starts with an initial value after which a configured delay time is followed. If a topology change event occurs the SPF is schedule after the time specified by the initial value, the device starts a timer for the time period specified by a configured hold time value. If no topology events occur during this hold time, the router returns to using the initial delay time.

If a topology event occurs during the hold time period, the next hold time period is recalculated to a value that is double the initial value. If no topology events occur during this extended hold time, the device resets to its initial value. If an event occurs during this extended hold time, the next hold time is doubled again. The doubling occurs as long as topology events occur during the calculated hold times until a configured maximum delay time value is reached or no event occurs (which resets the router to the initial hold time). The maximum value is then held until the hold time expires without a topology change event occurring. At any time that a hold time expires without a topology change event occurring, the router reverts to the initial hold value and begins the process all over again.

For example, if you set the initial delay timer to 100 milliseconds, the hold timer to 300 and the maximum hold timer to 2000 milliseconds, the following will occur:

If a topology change occurs the initial delay of 100 milliseconds will be observed. If a topology change occurs during the hold time of 300 milliseconds the hold time is doubled to 600 milliseconds. If a topology change event occurs during the 600 millisecond period, the hold time is doubled again to 1200 milliseconds. If a topology change event occurs during the 1200 millisecond period, the hold time is doubled to 2400 milliseconds. Because the maximum hold time is specified as 2000, the value will be held at 2000. This 2000 millisecond period will then repeat as long as topology events occur within the maximum 2000 millisecond hold time. When a maximum hold time expires without a topology event occurring, the router reverts to the initial delay time and the cycle repeats as described.

Therefore, longer SPF scheduling values can be used during network topology instability.

IETF RFC and internet draft support

The implementation of OSPF Graceful Restart supports the following IETF RFC:

- RFC 3623: Graceful OSPF Restart

NOTE

A secondary management module must be installed for the device to function as a graceful restart device. If the device functions as a graceful restart helper device only, there is no requirement for a secondary management module.

OSPFv2 non-stop routing

OSPFv2 can continue operation without interruption during hitless failover when the OSPFv2 non-stop routing (NSR) feature is enabled.

During graceful restart (GR), the restarting neighbors must help build routing information during a failover. However, GR may not be supported by all devices in a network. NSR eliminates this dependency.

NSR does not require support from neighboring devices to perform hitless failover, and OSPF can continue operation without interruption.

NOTE

NSR does not support IPv6-over-IPv4 tunneling and virtual links, so traffic loss is expected while performing hitless failover.

If the active management module fails, the standby management module takes over and maintains the current OSPF routes, link-state advertisements (LSAs), and neighbor adjacencies, so that there is no loss of existing traffic to the OSPF destination.

NOTE

NSR and Graceful Restart (GR) are mutually exclusive.

Limitations of NSR

- Configurations that occur before the switchover are lost due to the CLI synchronization.
- NSR does not support virtual links.
- Changes in the neighbor state or interface state before or during a switchover do not take effect.
- Traffic counters are not synchronized because the neighbor and LSA database counters are recalculated on the standby module during synchronization.
- LSA acknowledging is delayed because it has to wait until standby acknowledging occurs.
- Depending on the sequence of redistribution or new LSAs (from neighbors), the LSAs accepted within the limits of the database may change after switchover.

- In NSR hitless failover, after switchover, additional flooding-related protocol traffic is generated to the directly connected neighbors.
- OSPF startup timers, database overflow, and max-metric, are not applied during NSR switchover.
- Devices may generate OSPF log messages or reset OSPF neighbor timers, but these issues do not cause any OSPF or traffic disruption.

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-ospf-router)# nonstop-routing
```

The following example re-enables NSR for OSPFv2.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# nonstop-routing
```

Synchronization of critical OSPFv2 elements

All types of LSAs and the neighbor information are synchronized to the standby module using the NSR synchronization library and IPC mechanism to transmit and receive packets.

Link state database synchronization

To ensure non-stop routing, when the active management module fails the standby management module takes over from the active management module, with the identical OSPF link state database it had before the failure. The next shortest path first (SPF) run after the switchover yields the same result in routes as the active module had before the failure. The OSPF protocol requires that all devices in the network have identical databases.

LSA delayed acknowledging

When an OSPF device receives LSAs from its neighbor, it acknowledges the LSAs. After the acknowledgement is received, the neighbor removes this device from its retransmission list and stops resending the LSAs.

In the case of NSR, the device fails after receiving the LSA from its neighbor and acknowledges that neighbor upon receipt of an LSA. The LSA synchronization to the standby module is then completed. In this case the standby module, when taking over from the active module, does not have that LSA in its database and the already acknowledged neighbor does not retransmit that LSA. For this reason,

the NSR-capable device waits for LSA synchronization of the standby module to complete (Sync-Ack) before acknowledging the neighbor that sent the LSA.

LSA syncing and packing

When the LSA processing is completed on the active management module and the decision is made to install the LSA in its link state database (LSDB), OSPF synchronizes that LSA to the standby module. OSPF checks the current state of the database entry, whether or not it is marked for deletion. After checking the database state, OSPF packs the LSA status and other necessary information needed for direct installation in the standby OSPF LSDB, along with the LSA portion. When the LSA reaches the standby module, OSPF checks the database entry state in the buffer and takes appropriate action, such as adding, overwriting, updating, or deleting the LSA from the LSDB.

Neighbor device synchronization

When the neighbor device is added in the active management module, it is synchronized and added to the standby module. When the neighbor is deleted in the active module, it is synchronized to the standby module and deleted in the standby module. When the neighbor device state becomes 2way or full, the neighbor device is synchronized to the standby module. The following attributes of the neighbor device are synchronized to the standby module:

- Neighbor device ID
- Neighbor device IP address
- Destination device or backup destination device information
- Neighbor state 2way or full
- MD5 information
- Neighbor priority

Synchronization limitations

- If a neighbor device is inactive for 30 seconds, and if the standby module takes over in another 10 seconds, the neighbor device cannot be dropped. The inactivity timer starts again and takes another 40 seconds to drop the neighbor device.
- In standby module, the valid neighbor states are loading, down, 2way, and full. If the active management processor (MP) fails when the neighbor state is loading, the standby module cannot continue from loading, but the standby can continue from 2way and tries to establish adjacency between the neighboring devices.
- The minimum OSPF dead-interval timer value is 40 seconds. When the dead-interval value is configured to less than this minimum value, OSPF NSR cannot be supported.

Interface synchronization

Interface information is synchronized for interfaces such as PTPT, broadcast, and non-broadcast. Interface wait time is not synchronized to the standby module. If an interface waits for 30 seconds to determine the identity of the designated router (DR) or the backup designated router (BDR), and if the standby module takes over, the wait timer starts again and takes another 40 seconds for the interface state to change from waiting to BDR, DR, or DROther.

Standby module operations

The standby management module with OSPF configuration performs the following functions:

Neighbor database

Neighbor information is updated in the standby module based on updates from the active module. Certain neighbor state and interface transitions are synchronized to the standby module. By default, the neighbor timers on the standby module are disabled.

LSA database

The standby module processes LSA synchronization events from the active module and unpacks the LSA synchronization information to directly install it in its LSDB, as the LSA has already been processed on the active module. The information required to install all types of LSAs (and special LSAs such as Grace LSAs) is packed by OSPF on the active module in the synchronization buffer, so that you can directly install LSAs on the standby module without extra processing.

The standby module is not allowed to originate any LSAs of its own. This is to maintain all information consistently from the active module. The active module synchronizes self-originated LSAs to the standby module.

LSA aging is not applicable on the standby module. During synchronization from the active module, the current LSA age is recorded and the new database timestamp is created on the standby module to later derive the LSA age as needed.

When the active module sends the LSAs to the standby module, based on the message, the standby module deletes or updates its LSDB with the latest information.

LSA acknowledging or flooding are not done on the standby module. When the LSA synchronization update arrives from the active module, it will be directly installed into the LSDB.

OSPFv2 distribute list

A distribution list can be configured to explicitly deny specific routes from being eligible for installation in the IP route table. By default, all OSPFv2 routes in the OSPFv2 route table are eligible for installation in the IP route table. Receipt of LSAs are not blocked for the denied routes. The device still receives the routes and installs them in the OSPFv2 database. The denied OSPFv2 routes cannot be installed into the IP route table.

The OSPFv2 distribution list can be managed using ACLs or route maps to identify routes to be denied as described in the following sections:

- Configuring an OSPFv2 Distribution List using ACLs
- Configuring an OSPFv2 Distribution List using route maps

Configuring an OSPFv2 distribution list using ACLs

To configure an OSPFv2 distribution list using ACLs:

- Configure an ACL that identifies the routes you want to deny. Using a standard ACL allows you deny routes based on the destination network, but does not filter based on the network mask. To also filter based on the network mask of the destination network, use an extended ACL.
- Configure an OSPFv2 distribution list that uses the ACL as input.

Examples

In the following configuration example, the first three commands configure a standard ACL that denies routes to any 10.x.x.x destination network and allows all other routes for eligibility to be installed in the IP route table. The last three commands change the CLI to the OSPFv2 configuration level and configure an OSPFv2 distribution list that uses the ACL as input. The distribution list prevents routes to

any 10.x.x.x destination network from entering the IP route table. The distribution list does not prevent the routes from entering the OSPFv2 database.

```
device(config)# ip access-list standard no_ip
device(config-std-nacl)# deny 10.0.0.0 0.255.255.255
device(config-std-nacl)# permit any
device(config)# router ospf
device(config-ospf-router) # area 0
device(config-ospf-router) # distribute-list no_ip in
```

In the following example, the first three commands configure an extended ACL that denies routes to any 10.31.39.x destination network and allows all other routes for eligibility to be installed in the IP route table. The last three commands change the CLI to the OSPFv2 configuration level and configure an OSPFv2 distribution list that uses the ACL as input. The distribution list prevents routes to any 10.31.39.x destination network from entering the IP route table. The distribution list does not prevent the routes from entering the OSPFv2 database.

```
device(config)# ip access-list extended DenyNet39
device(config-ext-nacl)# deny ip 10.31.39.0 0.0.0.255 any
device(config-ext-nacl)# permit ip any any
device(config)# router ospf
device(config-ospf-router) # area 0
device(config-ospf-router) # distribute-list DenyNet39 in
```

In the following example, the first command configures a numbered ACL that denies routes to any 10.31.39.x destination network and allows all other routes for eligibility to be installed in the IP route table. The last three commands change the CLI to the OSPFv2 configuration level and configure an OSPF distribution list that uses the ACL as input. The distribution list prevents routes to any 10.31.39.x destination network from entering the IP route table. The distribution list does not prevent the routes from entering the OSPFv2 database.

```
device(config)# ip access-list 100 deny ip 10.31.39.0 0.0.0.255 any
device(config)# ip access-list 100 permit ip any any
device(config)# router ospf
device(config-ospf-router) # area 0
device(config-ospf-router) # distribute-list 100 in
```

Configuring an OSPFv2 distribution list using route maps

You can manage an OSPFv2 distribution list using route maps that apply match operations as defined by an ACL or an IP prefix list. You can also use other options available within the route maps and ACLs to further control the contents of the routes that OSPFv2 provides to the IP route table. This section describes an example of an OSPFv2 distribution list using a route map to specify an OSPFv2 administrative distance for routes identified by an IP prefix list.

To configure an OSPFv2 distribution list using route maps:

- Configure a route map that identifies the routes you want to manage
- Optionally configure an OSPFv2 administrative distance to apply to the OSPFv2 routes
- Configure an OSPFv2 distribution list that uses the route map as input

In the following example, the first two commands identify two routes using the **ip prefix-list test1** command. Next, a route map is created using the **prefix-list test1** command to identify the two routes and the **set distance** command to set the OSPFv2 administrative distance of those routes to 200. A distribution list is then configured under the OSPFv2 configuration that uses the route map named "setdistance" as input.

```
device(config)# ip prefix-list test1 seq 5 permit 10.0.0.2/32
device(config)# ip prefix-list test1 seq 10 permit 10.102.1.0/24
device(config)# route-map setdistance permit 1
device(config-routemap setdistance)# match ip address prefix-list test1
device(config-routemap setdistance)# set distance 200
device(config-routemap setdistance)# exit
device(config)# route-map setdistance permit 2
```

OSPFv2 route redistribution

```

device(config-routemap setdistance)# exit
device(config)# router ospf
device(config-ospf-router)# area 0
device(config-ospf-router)# area 1
device(config-ospf-router)# distribute-list route-map setdistance in
device(config-ospf-router)# exit

```

Once this configuration is implemented, the routes identified by the **ip prefix-list** command and matched in the route map will have their OSPFv2 administrative distance set to 200. This is displayed in the output from the **show ip route** command, as shown below.

```

device# show ip route
Total number of IP routes: 4
Type Codes - B:BGP D:Connected O:OSPF R:RIP S:Static; Cost - Dist/Metric
BGP Codes - i:iBGP e:eBGP
OSPF Codes - i:Inter Area 1:External Type 1 2:External Type 2

```

	Destination	Gateway	Port	Cost	Type	Uptime
1	10.0.0.2/32	10.1.1.2	ve 100	200/501	O	1h3m
2	10.102.1.0/24	10.1.1.2	ve 100	200/2	O	1h3m
3	10.102.6.0/24	10.1.1.2	ve 100	110/2	O	1h3m
4	10.102.8.0/30	DIRECT	ve 100	0/0	D	1h4m

Routes 1 and 2 demonstrate the actions of the example configuration as both display an OSPFv2 administrative distance value of 200. Note that the value is applied to both OSPFv2 learned routes that match the route-map instance containing the set distance clause. The other OSPFv2 route (route 3), which does not match the relevant instance, continues to have the default OSPFv2 administrative distance of 110.

OSPFv2 route redistribution

Route redistribution imports and translates different protocol routes into a specified protocol type. On the device, redistribution is supported for static routes, OSPF, RIP, and BGP. OSPF redistribution supports the import of static, RIP, and BGP routes into OSPF routes.

NOTE

The device advertises the default route into OSPF even if redistribution is not enabled, and even if the default route is learned through an IBGP neighbor. IBGP routes (including the default route) are not redistributed into OSPF by OSPF redistribution (for example, by the OSPF **redistribute** command).

In the figure below, the device acting as the ASBR (Autonomous System Boundary Router) can be configured between the RIP domain and the OSPF domain to redistribute routes between the two domains.

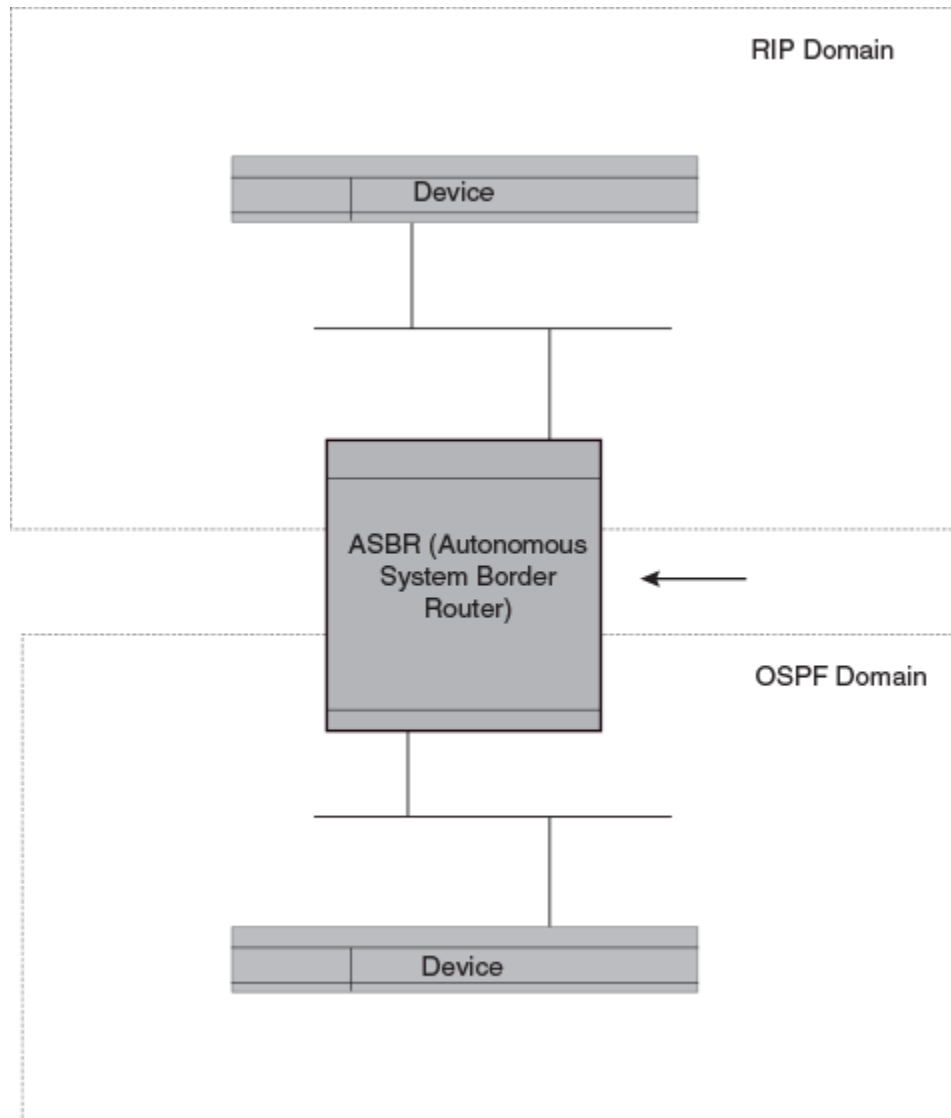
NOTE

The ASBR must be running both RIP and OSPF protocols to support this activity.

NOTE

Do not enable redistribution until you have configured the redistribution route map. Otherwise, you might accidentally overload the network with routes you did not intend to redistribute.

FIGURE 26 Redistributing OSPF and static routes to RIP routes



Redistributing routes into OSPFv2

OSPFv2 routes can be redistributed, and the routes to be redistributed can be specified.

The redistribution of RIP and static IP routes into OSPFv2 is configured on a device.

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

```
device# configure terminal
```

2. Enter the **router ospf** command to enter OSPFv2 router configuration mode and enable OSPFv2 on the device.

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

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

```
device(config-ospf-router)# redistribute static
```

4. Enter the **redistribute** command with the **rip** parameter to redistribute RIP routes.

```
device(config-ospf-router)# redistribute rip
```

The following example redistributes static and RIP routes into OSPFv2 on a device.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# redistribute static
device(config-ospf-router)# redistribute rip
```

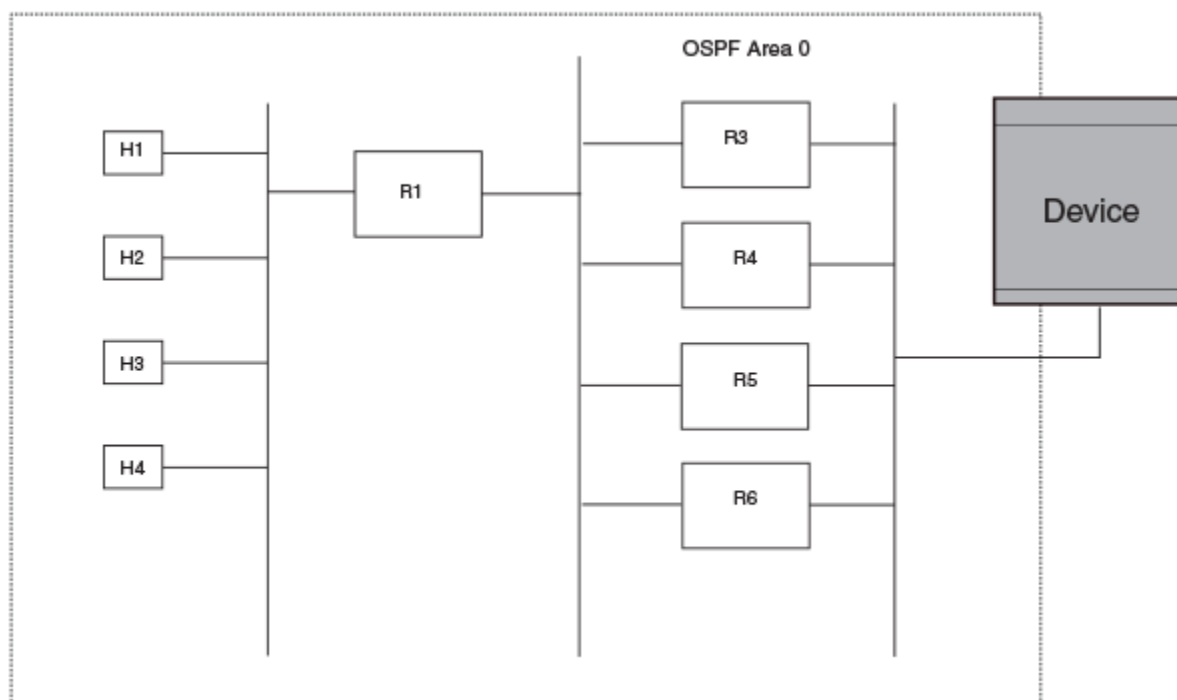
Load sharing

Brocade devices can load share among up to eight equal-cost IP routes to a destination. By default, IP load sharing is enabled. The default is 4 equal-cost paths but you can specify from 2 to 8 paths.

On the ICX 7750 device, the value range for the maximum number of load-sharing paths is from 2 through 32, which is controlled by the **system-max max-ecmp** command.

The device software can use the route information it learns through OSPF to determine the paths and costs.

FIGURE 27 Example OSPF network with four equal-cost paths



The device has four paths to R1:

- Router ->R3
- Router ->R4

- Router ->R5
- Router ->R6

Normally, the device chooses the path to the R1 with the lower metric. For example, if the metric for R3 is 1400 and the metric for R4 is 600, the device always chooses R4.

However, suppose the metric is the same for all four routers in this example. If the costs are the same, the device now has four equal-cost paths to R1. To allow the device to load share among the equal cost routes, enable IP load sharing. Four equal-cost OSPF paths are supported by default when you enable load sharing.

NOTE

The device is not source routing in these examples. The device is concerned only with the paths to the next-hop routers, not the entire paths to the destination hosts.

OSPF load sharing is enabled by default when IP load sharing is enabled.

Interface types to which the reference bandwidth does not apply

Some interface types are not affected by the reference bandwidth and always have the same cost regardless of the reference bandwidth in use:

- The cost of a loopback interface is always 1.
- The cost of a virtual link is calculated using the Shortest Path First (SPF) algorithm and is not affected by the auto-cost feature.
- The bandwidth for tunnel interfaces is 9 Kbps and is also subject to the auto-cost reference bandwidth setting.

Changing the reference bandwidth for the cost on OSPFv2 interfaces

Each interface on which OSPFv2 is enabled has a cost associated with it. The device advertises its interfaces and their costs to OSPFv2 neighbors. For example, if an interface has an OSPFv2 cost of ten, the device advertises the interface with a cost of ten to other OSPFv2 routers.

By default, an interface's OSPFv2 cost is based on the port speed of the interface. The cost is calculated by dividing the reference bandwidth by the port speed. The default reference bandwidth is 100 Mbps, which results in the following default costs:

- 10 Mbps port - 10
- All other port speeds - 1

You can change the reference bandwidth. The following formula is used to calculate the cost:

Cost = reference-bandwidth/interface-speed

If the resulting cost is less than 1, the cost is rounded up to 1. The default reference bandwidth results in the following costs:

- 10 Mbps port's cost = $100/10 = 10$
- 100 Mbps port's cost = $100/100 = 1$
- 1000 Mbps port's cost = $100/1000 = 0.10$, which is rounded up to 1
- 10 Gbps port's cost = $100/10000 = 0.01$, which is rounded up to 1

The bandwidth for interfaces that consist of more than one physical port is calculated as follows:

- LAG group - The combined bandwidth of all the ports.
- Virtual interface - The combined bandwidth of all the ports in the port-based VLAN that contains the virtual interface.

The default reference bandwidth is 100 Mbps. You can change the reference bandwidth to a value from 1–4294967.

If a change to the reference bandwidth results in a cost change to an interface, the device sends a link-state update to update the costs of interfaces advertised by the device.

NOTE

If you specify the cost for an individual interface, the cost you specify overrides the cost calculated by the software.

OSPFv2 over VRF

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-ospf-router-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.

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 **max-metric router-lsa** command with the **on-startup** keyword and specify a value to specify a period of time to advertise a maximum metric after a restart before advertising with a normal metric.

```
device(config-ospf-router)# max-metric router-lsa on-startup 85
```

The following example configures an OSPFv2 device to advertise a maximum metric for 85 seconds after a restart before advertising with a normal metric.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# max-metric router-lsa on-startup 85
```

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-ospf-router)# rfc1583-compatibility
```

The following example re-enables OSPFv2 compatibility with RFC 1583.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# rfc1583-compatibility
```

Changing default settings

Refer to the FastIron *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-ospf-router)# no log all
```

The following example re-enables the logging of all OSPFv2 events.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# log all
```

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 startup configuration file after disabling one of these protocols, all the configuration information for the disabled protocol is removed from the startup configuration file.
- If you have disabled the protocol but have not yet saved the configuration to the startup configuration file and reloaded the software, you can restore the configuration information by re-entering the **router ospf** command, or by selecting the Web management option to enable the protocol. If you have already saved the configuration to the startup configuration file and reloaded the software, the information is gone.
- If you are testing an OSPFv2 configuration and are likely to disable and re-enable the protocol, you might want to make a backup copy of the startup 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 SPF  0           (msecs)
Maximum hold time for SPF  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

Displaying OSPFv2 results

8. 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!		

OSPFv3

• OSPFv3 overview.....	263
• Configuring the router ID.....	264
• Enabling OSPFv3.....	264
• Configuring OSPFv3.....	264
• OSPFv3 areas.....	265
• Virtual links.....	270
• OSPFv3 route redistribution.....	273
• Default route origination.....	275
• Disabling and re-enabling OSPFv3 event logging.....	275
• Filtering OSPFv3 routes.....	276
• SPF timers.....	279
• OSPFv3 administrative distance.....	280
• Changing the reference bandwidth for the cost on OSPFv3 interfaces.....	281
• OSPFv3 LSA refreshes.....	282
• External route summarization.....	283
• OSPFv3 over VRF.....	283
• Setting all OSPFv3 interfaces to the passive state.....	285
• OSPFv3 graceful restart helper.....	286
• OSPFv3 non-stop routing.....	287
• IPsec for OSPFv3.....	288
• Displaying OSPFv3 results.....	293

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, Brocade 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.

Before enabling the device to run OSPFv3, you must perform the following steps:

- Enable the forwarding of IPv6 traffic on the device using the **ipv6 unicast-routing** command.
 - Enable IPv6 on each interface on which you plan to enable OSPFv3. You enable IPv6 on an interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.
1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **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-ospf6-router)#
```

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 32 ranges in an OSPF 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-ospf6-router)# area 0
```

5. Enter the **area** command to define a second OSPFv3 area ID.

```
device(config-ospf6-router)# area 10.1.1.1
```

The following example assigns an OSPFv3 ID to two areas. One of the areas is assigned by decimal number. The second area is assigned by IP address.

```
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 router ospf
device(config-ospf6-router)# area 0
device(config-ospf6-router)# area 10.1.1.1
```

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 ve 1
```

3. Enter the **ipv6 address** command to add an IPv6 address to the interface.

```
device(config-vif-1)# ipv6 address 2001:db8:93e8:cc00::1
```

4. Enter the **ipv6 ospf area** command.

```
device(config-vif-1)# ipv6 ospf area 0
```

Area 0 is assigned to the specified interface with the IPv6 address of 2001:db8:93e8:cc00::1.

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

```
device(config-vif-1)# exit
```

6. Enter the **interface** command and specify an interface.

```
device(config)# interface ve 2
```

7. Enter the **ipv6 address** command to add an IPv6 address to the interface.

```
device(config-vif-2)# ipv6 address 2001:db8:93e8:cc00::2
```

8. Enter the **ipv6 ospf area** command.

```
device(config-vif-2)# ipv6 ospf area 1
```

Area 1 is assigned to the specified interface with the IPv6 address of 2001:db8:93e8:cc00::1.

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 ve 1
device(config-vif-1)# ipv6 address 2001:db8:93e8:cc00::1
device(config-vif-1)# ipv6 ospf area 0
device(config-vif-1)# exit
device(config)# interface ve 2
device(config-vif-2)# ipv6 address 2001:db8:93e8:cc00::2
device(config-vif-2)# ipv6 ospf area 1
```

Stub area and totally stubby area

A stub area is an area in which advertisements of external routes are not allowed, reducing the size of the database. A totally stubby area (TSA) is a stub area in which summary link-state advertisement (type 3 LSAs) are not sent. A default summary LSA, with a prefix of 0.0.0.0/0 is originated into the stub area by an ABR, so that devices in the area can forward all traffic for which a specific route is not known, via ABR.

A stub area disables advertisements of external routes. By default, the ABR sends summary LSAs (type 3 LSAs) into stub areas. You can further reduce the number of LSAs sent into a stub area by configuring the device to stop sending type 3 LSAs into the area. You can disable the summary LSAs to create a TSA when you are configuring the stub area or after you have configured the area.

The ABR of a totally stubby area disables origination of summary LSAs into this area, but still accepts summary LSAs from OSPF neighbors and floods them to other neighbors.

When you enter the **area stub** command with the **no-summary** keyword and specify an area to disable the summary LSAs, the change takes effect immediately. If you apply the option to a previously configured area, the device flushes all the summary LSAs it has generated (as an ABR) from the area with the exception of the default summary LSA originated. This default LSA is needed for the internal routers, since external routes are not propagated to them.

NOTE

Stub areas and TSAs apply only when the device is configured as an Area Border Router (ABR) for the area. To completely prevent summary LSAs from being sent to the area, disable the summary LSAs on each OSPF router that is an ABR for the area.

Assigning 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-ospf6-router)# area 4 stub 100
```

Area 4 is defined as a stub area with an additional cost of 100.

The following example sets an additional cost of 100 on a stub area defined as 4.

```
device# configure terminal
device(config)# ip router-id 10.4.4.4
device(config)# ipv6 router ospf
device(config-ospf6-router)# area 4 stub 100
```

Not-so-stubby area

A not-so-stubby-area (NSSA) is an OSPFv3 area that provides the benefits of stub areas with the extra capability of importing external route information. OSPFv3 does not flood external routes from other areas into an NSSA, but does translate and flood route information from the NSSA into other areas such as the backbone.

NSSAs are especially useful when you want to aggregate type 5 External LSAs (external routes) before forwarding them into an OSPFv3 area. When you configure an NSSA, you can specify an address range for aggregating the external routes that the ABR of the NSSAs exports into other areas.

The OSPFv3 specification (RFC 2740) prohibits the advertising of type 5 LSAs and requires OSPFv3 to flood type 5 LSAs throughout a routing domain.

If the router is an ABR, you can prevent any type 3 and type 4 LSA from being injected into the area by configuring a nssa with the **no-summary** parameter. The only exception is that a default route is injected into the NSSA by the ABR, and strictly as a type 3 LSA. The default type 7 LSA is not originated in this case.

By default, the device's NSSA translator role is set to candidate and the router participates in NSSA translation election, if it is an ABR. You can also configure the NSSA translator role.

In the case where an NSSA ABR is also an ASBR, the default behavior is that it originates type 5 LSAs into normal areas and type 7 LSAs into an NSSA. But you can prevent an NSSA ABR from generating type 7 LSAs into an NSSA by configuring the **no-redistribution** parameter.

Configuring an NSSA

OSPFv3 areas can be defined as NSSA areas with configurable parameters.

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

```
device# configure terminal
```

2. Enter the **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-ospf6-router)# area 3 nssa default-information-originate metric 33
```

Area 3 is defined as an NSSA with the default route option and an additional cost of 33.

The following example sets an additional cost of 33 on an NSSA defined as 3.

```
device# configure terminal
device(config)# ip router-id 10.3.3.3
device(config)# ipv6 router ospf
device(config-ospf6-router)# area 3 nssa default-information-originate metric 33
```

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.

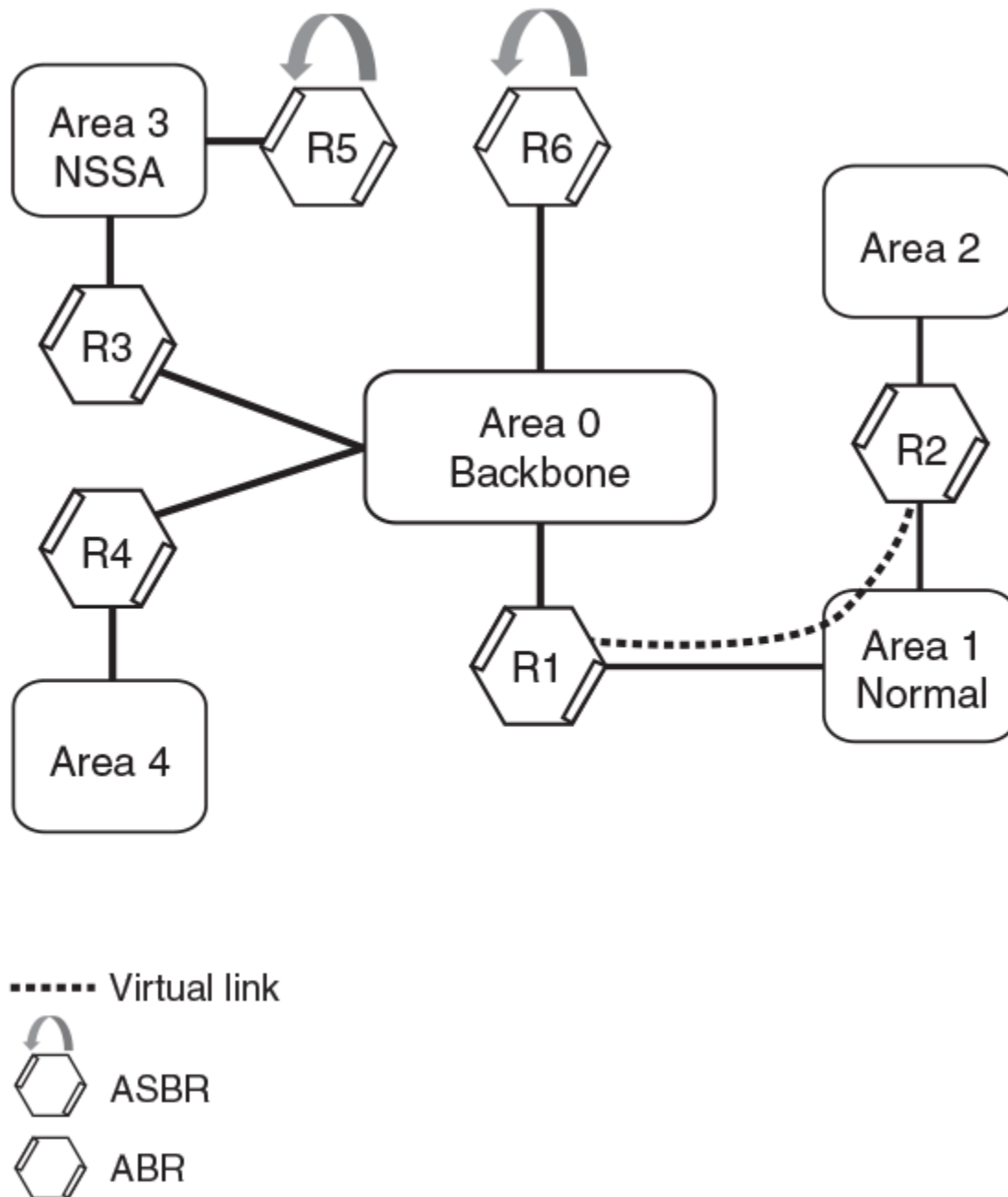
Virtual links

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 28 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-ospf6-router)# area 0
```

5. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ospf6-router)# 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-ospf6-router)# area 1 virtual-link 10.2.2.2
```


- On ABR2, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

- Enter the **ip router-id** command to specify the router ID.

```
device(config)# ip router-id 10.2.2.2
```

- Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

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

- Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ospf6-router)# area 1
```

- Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ospf6-router)# area 2
```

- Enter the **area virtual-link** command and the ID of the OSPFv3 device at the remote end of the virtual link to configure the virtual link endpoint.

```
device(config-ospf6-router)# area 1 virtual-link 10.1.1.1
```

The following example configures a virtual link between two devices.

```
ABR1:
device1# configure terminal
device1(config)# ip router-id 10.1.1.1
device1(config)# ipv6 router ospf
device1(config-ospf6-router)# area 0
device1(config-ospf6-router)# area 1
device1(config-ospf6-router)# area 1 virtual-link 10.2.2.2
```

```
ABR2:
device2# configure terminal
device2(config)# ip router-id 10.2.2.2
device2(config)# ipv6 router ospf
device2(config-ospf6-router)# area 1
device2(config-ospf6-router)# area 2
device2(config-ospf6-router)# area 1 virtual-link 10.1.1.1
```

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+
- RIPng

You can redistribute routes in the following ways:

- By route types. For example, the device redistributes all IPv6 static routes.
- By using a route map to filter which routes to redistribute. For example, the device redistributes specified IPv6 static routes only.

NOTE

You must configure the route map before you configure a redistribution filter that uses the route map.

NOTE

For an external route that is redistributed into OSPFv3 through a route map, the metric value of the route remains the same unless the metric is set by the **set metric** command inside the route map or the **default-metric** command. For a route redistributed without using a route map, the metric is set by the metric parameter if set or the **default-metric** command if the metric parameter is not set.

Redistributing routes into OSPFv3

OSPFv3 routes can be redistributed, and the routes to be redistributed can be specified.

The redistribution of static 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-ospf6-router)# redistribute static
```

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

```
device# configure terminal
```

5. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

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

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

```
device(config-ospf6-router)# redistribute connected route-map rmap1
```

The following example redistributes static routes into OSPFv3 on a device.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# redistribute static
```

The following example redistributes connected routes into OSPFv3 on a device and specifies a route map.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# redistribute connected route-map rmap1
```

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-ospf6-router)# default-information-originate always metric 2 metric-type type1
```

A default type 1 external route with a metric of 2 is created and advertised.

The following example creates and advertises a default route with a metric of 2 and a type 1 external route.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# default-information-originate always metric 2 metric-type type1
```

Disabling and re-enabling OSPFv3 event logging

OSPFv3 event logging, such as neighbor state changes and database overflow conditions, can be disabled and re-enabled.

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

```
device# configure terminal
```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

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

- Enter the **no log-status-change** command to disable the logging of OSPFv3 events.

```
device(config-ospf6-router)# no log-status-change
```

The following example re-enables the logging of OSPFv3 events.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# log-status-change
```

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. However, unlike OSPFv2 distribution lists, which filter routes based on criteria specified in an Access Control List (ACL), OSPFv3 distribution lists can filter routes using information specified in an IPv6 prefix list or a route map.

Configuring an OSPFv3 distribution list using an IPv6 prefix list as input

An IPv6 prefix list can be used to filter OSPFv3 routes.

- Enter the **show ipv6 ospf route** command to verify the OSPFv3 routes.

```
device> show ipv6 ospf route
Current Route count: 5
Intra: 3 Inter: 0 External: 2 (Type1 0/Type2 2)
Equal-cost multi-path: 0
Destination      Options   Area      Cost Type2 Cost
Next Hop Router  Outgoing Interface
*IA 2001:db8:1::/64 ----- 10.0.0.1      0 0
::              ve 10
*E2 2001:db8:2::/64 ----- 0.0.0.0      10 0
fe80::2e0:52ff:fe00:10 ve 10
*IA 2001:db8:3::/64 V6E---R-- 0.0.0.0      11 0
fe80::2e0:52ff:fe00:10 ve 10
*IA 2001:db8:4::/64 ----- 0.0.0.0      10 0
::              ve 11
*E2 2001:db8:5::/64 ----- 0.0.0.0      10 0
fe80::2e0:52ff:fe00:10 ve 10
```

- Enter the **enable** command to access privileged EXEC mode.

```
device> enable
```

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

```
device# configure terminal
```

- Enter the **ip router-id** command to specify the router ID.

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

- Enter the **ipv6 prefix-list** command, using the **deny** keyword and specify a name.

```
device(config)# ipv6 prefix-list filterOspfRoutes seq 5 deny 2001:db8:2::/642
```

An IPv6 prefix list called “filterOspfRoutes” that denies route 2001:db8:2::/64 is configured.

6. Enter the **ipv6 prefix-list** command using the **deny** keyword and specify a name. Use the **ge** keyword to specify a prefix length greater than or equal to the *ipv6-prefix/prefix-length* arguments. Use the **le** keyword to specify a prefix length less than or equal to the *ipv6-prefix/prefix-length* arguments.

```
device(config)# ipv6 prefix-list filterOspfRoutes seq 7 permit ::/0 ge 1 le 128
```

7. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

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

8. Enter the **distribute-list prefix-list** command, using the **in** keyword and specifying a name to configure a distribution list that applies the filterOspfRoutes prefix list globally.

```
device(config-ospf6-router)# distribute-list prefix-list filterOspfRoutes in
```

9. Enter the **exit** command until you return to user EXEC mode.

```
device(config-ospf6-router)# exit
```

10. Enter the **show ipv6 ospf route** command to verify that route 2001:db8:2::/64 is now omitted from the route table.

```
device> show ipv6 ospf route
Current Route count: 4
Intra: 3 Inter: 0 External: 1 (Type1 0/Type2 1)
Equal-cost multi-path: 0
Destination      Options  Area      Cost Type2 Cost
Next Hop Router  Outgoing Interface
*IA 2001:db8:1::/64 ----- 10.0.0.1      0 0
::              ve 10
*IA 2001:db8:3::/64 V6E---R-- 0.0.0.0      11 0
fe80::2e0:52ff:fe00:10 ve 10
*IA 2001:db8:4::/64 ----- 0.0.0.0      10 0
::              ve 11
*E2 2001:db8:5::/64 ----- 0.0.0.0      10 0
fe80::2e0:52ff:fe00:10 ve 10
```

The following example configures an IPv6 prefix list that is used to filter OSPFv3 routes. A distribution list is then configured and route 2001:db8:2::/64 is omitted from the OSPFv3 route table.

```
device> show ipv6 ospf route
device> enable
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 prefix-list filterOspfRoutes seq 5 deny 2001:db8:2::/642
device(config)# ipv6 prefix-list filterOspfRoutes seq 7 permit ::/0 ge 1 le 128
device(config)# ipv6 router ospf
device(config-ospf6-router)# distribute-list prefix-list filterOspfRoutes in
device(config-ospf6-router)# exit
device> show ipv6 ospf route
```

Configuring an OSPFv3 distribution list using a route map as input

A route map that matches internal routes can be used to filter OSPFv3 routes.

1. Enter the **show ipv6 ospf route** command to verify the OSPFv3 routes.

```
device# show ipv6 ospf route
Current Route count: 5
Intra: 3 Inter: 0 External: 2 (Type1 0/Type2 2)
Equal-cost multi-path: 0
Destination          Options   Area          Cost Type2 Cost
Next Hop Router      Outgoing  Interface
*IA 2001:db8:1::/64   -----  10.0.0.1       0  0
::                  ve 10
*E2 2001:db8:2::/64   -----  0.0.0.0       10 0
fe80::2e0:52ff:fe00:10 ve 10
*IA 2001:db8:3::/64   V6E---R-- 0.0.0.0       11 0
fe80::2e0:52ff:fe00:10 ve 10
*IA 2001:db8:4::/64   -----  0.0.0.0       10 0
::                  ve 11
*E2 2001:db8:5::/64   -----  0.0.0.0       10 0
fe80::2e0:52ff:fe00:10 ve 10
```

2. Enter the **enable** command to access privileged EXEC mode.

```
device> enable
```

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

```
device# configure terminal
```

4. Enter the **ip router-id** command to specify the router ID.

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

5. Enter the **route-map** command, using the **permit** keyword and specify a name.

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

A route map called "allowInternalRoutes" that permits a matching pattern is configured.

6. Enter the **match route-type** command, using the **internal** keyword to match internal route types in the route map instance.

```
device(config-routemap allowInternalRoutes)# match route-type internal
```

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

```
device(config-routemap allowInternalRoutes)# exit
```

8. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

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

9. Enter the **distribute-list route-map** command, using the **in** keyword and specifying a name to create a distribution list a distribution list using the configured route map "allowinternalroutes".

```
device(config-ospf6-router)# distribute-list route-map allowinternalroutes in
```

10. Enter the **exit** command until you return to user EXEC mode.

```
device(config-ospf6-router)# exit
```

11. Enter the **show ipv6 ospf route** command to verify that the external routes are omitted from the OSPFv3 route table.

```
device> show ipv6 ospf route
Current Route count: 3
Intra: 3 Inter: 0 External: 0 (Type1 0/Type2 0)
Equal-cost multi-path: 0
Destination          Options   Area      Cost Type2 Cost
Next Hop Router      Outgoing  Interface
-----
*IA 2001:db8:3001::/64 ----- 10.0.0.1      0  0
::                  ve 10
*IA 2001:db8:3015::/64 V6E---R-- 0.0.0.0      11 0
fe80::2e0:52ff:fe00:10 ve 10
*IA 2001:db8:3020::/64 ----- 0.0.0.0      10 0
::                  ve 11
```

The following example configures a route map that is used to filter OSPFv3 routes. A distribution list is then configured and external routes omitted from the OSPFv3 route table.

```
device> show ipv6 ospf route
device> enable
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# route-map allowInternalRoutes permit 10
device(config-route-map allowInternalRoutes)# match route-type internal
device(config-route-map allowInternalRoutes)# exit
device(config)# ipv6 router ospf
device(config-ospf6-router)# distribute-list route-map allowinternalroutes in
device(config-ospf6-router)# exit
device> show ipv6 ospf route
```

SPF timers

The device uses an SPF delay timer and an SPF hold-time timer to calculate the shortest path for OSPFv3 routes. The values for both timers can be changed.

The device uses the following timers when calculating the shortest path for OSPFv3 routes:

- **SPF delay:** When the device receives a topology change, it waits before starting a Shortest Path First (SPF) calculation. By default, the device waits 5 seconds. You can configure the SPF delay to a value from 0 through 65535 seconds. If you set the SPF delay to 0 seconds, the device immediately begins the SPF calculation after receiving a topology change.
- **SPF hold time:** The device waits a specific amount of time between consecutive SPF calculations. By default, it waits 10 seconds. You can configure the SPF hold time to a value from 0 through 65535 seconds. If you set the SPF hold time to 0 seconds, the device does not wait between consecutive SPF calculations.

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-ospf6-router)# timers spf 1 5
```

The SPF delay is changed to 1 second and the SPF hold time is changed to 5 seconds.

The following example changes the SPF delay and hold time.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# timers spf 1 5
```

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 Brocade 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-ospf6-router)# 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-ospf6-router)# 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-ospf6-router)# distance external 100
```

The administrative distance for external routes is changed from the default to 100.

The following example changes the default administrative distances for intra-area routes, inter-area routes, and external routes.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# distance intra-area 80
device(config-ospf6-router)# distance inter-area 90
device(config-ospf6-router)# distance external 100
```

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-ospf6-router)# 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-ospf6-router)# 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-ospf6-router)# timers lsa-group-pacing 120
```

The OSPFv3 LSA pacing interval is changed to 120 seconds (two minutes).

The following example restores the pacing interval to the default value of 240 seconds (4 minutes).

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# no timers lsa-group-pacing
```

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.

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

```
device# configure terminal
```

2. Enter the **vrf** command and specify a name to enter Virtual Routing and Forwarding (VRF) configuration mode and create a non-default VRF instance.

```
device(config)# vrf green
```

3. Enter the **rd** command, assigning an administrative number and arbitrary number the route, to distinguish a route for VRF green.

```
device(config-vrf-green)# rd 100:200
```

4. Enter the **ip router-id** command to specify the router ID.

```
device(config-vrf-green)# ip router-id 10.11.12.14
```

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

```
device(config-vrf-green)# address-family ipv6
```

6. Enter the **exit** command until you return to global configuration mode.

```
device(config-vrf-green-ipv6)# exit
```

7. Enter the **ipv6 router ospf** command and specify a VRF name to enter OSPFv3 router VRF configuration mode and enable OSPFv3 on a non-default VRF.

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

The following example enables OSPFv3 in a non-default VRF.

```
device# configure terminal
device(config)# vrf green
device(config-vrf-green)# rd 100:200
device(config-vrf-green)# ip router-id 10.11.12.14
device(config-vrf-green)# address-family ipv6
device(config-vrf-green-ipv6)#
device(config-vrf-green-ipv6)# exit
device(config)# ipv6 router ospf vrf green
device(config-ospf6-router-vrf-green)#
```

Assigning OSPFv3 areas in a non-default VRF

Areas can be assigned as OSPFv3 areas in a non-default VRF.

Enable IPv6 on each interface on which you plan to enable OSPFv3. You enable IPv6 on an interface by configuring an IP address or explicitly enabling IPv6 on that interface.

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

```
device# configure terminal
```

2. Enter the **vrf** command and specify a name to enter Virtual Routing and Forwarding (VRF) configuration mode and create a non-default VRF instance.

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

3. Enter the **rd** command, assigning an administrative number and arbitrary number the route, to distinguish a route for VRF green.

```
device(config-vrf-red)# rd 100:200
```

4. Enter the **ip router-id** command to specify the router ID.

```
device(config-vrf-red)# ip router-id 10.11.12.14
```

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

```
device(config-vrf-red)# address-family ipv6
```

6. Enter the **exit** command until you return to global configuration mode.

```
device(config-vrf-red-ipv6)# exit
```

7. Enter the **ipv6 router ospf** command and specify a VRF name to enter OSPFv3 configuration mode and enable OSPFv3 in a non-default VRF.

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

8. Enter the **area** command to define an OSPFv3 area ID.

```
device(config-ospf6-router-vrf-red)# area 0
```

9. Enter the **area** command to define a second OSPFv3 area ID.

```
device(config-ospf6-router-vrf-red)# area 10.1.1.1
```

The following example assigns an OSPFv3 ID to two areas in a non-default VRF instance. One of the areas is assigned by decimal number. The second area is assigned by IP address.

```
device# configure terminal
device(config)# vrf red
device(config-vrf-red)# rd 100:200
device(config-vrf-red)# ip router-id 10.11.12.13
device(config-vrf-red)# address-family ipv6
device(config-vrf-red-ipv6)#
device(config-vrf-red-ipv6)# exit
device(config)# ipv6 router ospf vrf red
device(config-ospf6-router-vrf-red)# area 0
device(config-ospf6-router-vrf-red)# area 10.1.1.1
```

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-ospf6-router)# default-passive-interface
```

The following example sets all OSPFv3 interfaces as passive, causing them to drop all the OSPFv3 control packets.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# default-passive-interface
```

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.

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 with the **strict-lsa-checking** to disable the GR helper with strict link-state advertisement (LSA) checking.

```
device(config-ospf6-router)# no graceful-restart helper strict-lsa-checking
```

The following example disables the GR helper with strict link-state advertisement (LSA) checking.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# no graceful-restart helper strict-lsa-checking
```

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-ospf6-router)# graceful-restart helper strict-lsa-checking
```

The following example re-enables the GR helper with strict LSA checking.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# graceful-restart helper strict-lsa-checking
```

OSPFv3 non-stop routing

OSPFv3 can continue operation without interruption during hitless failover when the NSR feature is enabled.

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

NSR does not support IPv6-over-IPv4 tunnels and 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-ospf6-router)# nonstop-routing
```

The following example re-enables NSR for OSPFv3.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ospf6-router)# nonstop-routing
```

IPsec for OSPFv3

IP Security (IPsec) secures OSPFv3 communications by authenticating and encrypting each IP packet of a communication session.

IPsec provides security features such as data integrity, replay protection, and message confidentiality. You can use IPsec to secure specific OSPFv3 areas and interfaces and protect OSPFv3 virtual links.

The Encapsulating Security Payload (ESP) protocol authenticates routing information between peers. ESP can provide message confidentiality, connectionless data integrity, and optional replay protection. ESP has both a header and a trailer. The authentication data of ESP cannot protect the outer IP header, only the payload that is being encrypted.

IPsec 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.

Brocade devices support the following components of IPsec for IPv6-addressed packets:

- Authentication through ESP in transport mode
- Hashed Message Authentication Code-Secure Hash Algorithm 1 (HMAC-SHA-1) as the authentication algorithm
- Security parameter index (SPI)
- Manual configuration of keys
- Configurable rollover timer

IPsec can be enabled on the following logical entities:

- Interface
- Area
- Virtual link

IPsec is based on security associations (SAs). With respect to traffic classes, this implementation of IPsec uses a single security association between the source and destination to support all traffic classes and does not differentiate between the different classes of traffic that the DSCP bits define.

IPsec on a virtual link is a global configuration. Interface and area IPsec configurations are more granular.

Among the entities that can have IPsec protection, the interfaces and areas can overlap. The interface IPsec configuration takes precedence over the area IPsec configuration when an area and an interface within that area use IPsec. Therefore, if you configure IPsec for an interface and an area configuration also exists that includes this interface, the interface's IPsec configuration is used by that interface. However, if you disable IPsec on an interface, IPsec is disabled on the interface even if the interface has its own specific authentication.

For IPsec, the system generates two types of databases. The Security Association Database (SAD) contains a security association for each interface or one global database for a virtual link. Even if IPsec is configured for an area, each interface that uses the area's IPsec still has its own security association in the SAD. Each SA in the SAD is a generated entry that is based on your specifications of an authentication protocol (for example, ESP), destination address, and a security parameter index (SPI). The SPI number is user-specified according to the network plan. Consideration for the SPI values to specify must apply to the whole network.

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:

- ESP protocol
- Authentication
- Hashed Message Authentication Code-Secure Hash Algorithm 1 (HMAC-SHA-1) authentication algorithm
- Security parameter index (SPI)
- A 40-character key using hexadecimal characters
- An option for not encrypting the keyword when it appears in **show** command output
- Key rollover timer
- Specifying the key add remove timer

IPsec for OSPFv3 considerations

IPsec generates security associations and security policies based on certain user-specified parameters. Refer to the *FastIron Command Reference* for more information on user-specified parameters.

- The system creates a security association for each interface or virtual link based on the values specified by the user.
- The system creates a security policy database for each interface or virtual link based on the values specified by the user.
- You can configure the same SPI and key on multiple interfaces and areas, but they still have unique IPsec configurations because the SA and policies are added to each separate security policy database (SPD) that is associated with a particular interface. If you configure an SA with the same SPI in multiple places, the rest of the parameters associated with the SA—such as key, cryptographic algorithm, security protocol, and so on—must match. If the system detects a mismatch, it displays an error message.
- IPsec authentication for OSPFv3 requires the use of multiple SPDs, one for each interface. A virtual link has a separate, global SPD. The authentication configuration on a virtual link must be different from the authentication configuration for an area or interface, as required by RFC 4552. The interface number is used to generate a non-zero security policy database identifier (SPDID), but for the global SPD for a virtual link, the system-generated SPDID is always zero. As a hypothetical example, the SPD for interface eth 1/1/1 might have the system-generated SPDID of 1, and so on.
- If you change an existing key, you must also specify a different SPI value. For example, in an interface context where you intend to change a key, you must enter a different SPI value—which occurs before the key parameter on the command line—before you enter the new key.
- The old key is active for twice the current configured key rollover interval for the inbound direction. In the outbound direction, the old key remains active for a duration equal to the key rollover interval. If the key rollover interval is set to 0, the new key immediately takes effect for both directions.

Configuring IPsec on an OSPFv3 area

IPsec can be configured to secure communications on an OSPFv3 area.

Currently certain keyword parameters must be entered though only one keyword choice is possible for that parameter. For example, the only authentication algorithm is HMAC-SHA1-96, but you must nevertheless enter the **sha1** keyword for this algorithm. Also, although ESP is currently the only authentication protocol, you must enter the **esp** keyword.

NOTE

When IPsec is configured for an area, the security policy is applied to all the interfaces in the area.

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

```
device# configure terminal
```

2. Enter the **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 ipsec spi spi esp sha1**, specifying an area, and enter a 40-character hexadecimal key.

```
device(config-ospf6-router)# area 0 authentication ipsec spi 600 esp sha1
abcef12345678901234fedcba098765432109876
```

IPsec is configured in OSPFv3 area 0 with a security parameter index (SPI) value of 600, and Hashed Message Authentication Code (HMAC) Secure Hash Algorithm 1 (SHA-1) authentication is enabled.

The following example enables HMAC SHA-1 authentication for the OSPFv3 area, setting an SPI value of 600.

```
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 router ospf
device(config-ospf6-router)# area 0 authentication ipsec spi 600 esp sha1
abcef12345678901234fedcba098765432109876
```

The following example enables AH and MD5 authentication for the OSPFv3 area, setting an SPI value of 600.

Configuring IPsec on an OSPFv3 interface

IPsec can be configured to secure communications on an OSPFv3 interface.

For IPsec to work, the IPsec configuration must be the same on all the routers to which an interface connects.

Currently certain keyword parameters must be entered though only one keyword choice is possible for that parameter. For example, the only authentication algorithm is HMAC-SHA1-96, but you must nevertheless enter the **sha1** keyword for this algorithm. Also, although ESP is currently the only authentication protocol, you must enter the **esp** keyword.

NOTE

Ensure that OSPFv3 areas are assigned. All device interfaces must be assigned to one of the defined areas on an OSPFv3 router. When an interface is assigned to an area, all corresponding subnets on that interface are automatically included in the assignment.

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

```
device# configure terminal
```

2. Enter the **interface** command and specify an interface.

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

3. Enter the **ipv6 ospf area** command to assign a specified area to the interface.

```
device(config-vif-1)# ipv6 ospf area 0
```

4. Enter **ipv6 ospf authentication ipsec spi value esp sha1** and specify a 40-character hexadecimal key.

```
device(config-vif-1)# ipv6 ospf authentication ipsec spi 512 esp sha1
abcef12345678901234fedcba098765432109876
```

IPsec is configured on the specified interface with a security parameter index (SPI) value of 512, and the Encapsulating Security Payload (ESP) protocol is selected. Secure Hash Algorithm 1 (SHA-1) authentication is enabled.

The following example enables ESP and SHA-1 on a specified OSPFv3 virtual Ethernet (VE) interface.

```
device# configure terminal
device(config)# interface ve 1
device(config-vif-1)# ipv6 ospf area 0
device(config-vif-1)# ipv6 ospf authentication ipsec spi 512 esp sha1
abcef12345678901234fedcba098765432109876
```

Configuring IPsec on OSPFv3 virtual links

IP Security (IPsec) can be configured for virtual links.

An OSPFv3 virtual link must be configured.

Currently certain keyword parameters must be entered though only one keyword choice is possible for that parameter. For example, the only authentication algorithm is HMAC-SHA1-96, but you must nevertheless enter the **sha1** keyword for this algorithm. Also, although ESP is currently the only authentication protocol, you must enter the **esp** keyword.

The virtual link IPsec security associations (SAs) and policies are added to all interfaces of the transit area for the outbound direction. For the inbound direction, IPsec SAs and policies for virtual links are added to the global database.

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

```
device# configure terminal
```

2. Enter the **ip router-id** command to specify the router ID.

```
device(config)# ip router-id 10.1.1.1
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 configuration mode and enable OSPFv3 on the device.

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

4. Enter **area virtual-link authentication ipsec spi value esp sha1 no-encrypt key**, specifying an area address and the ID of the OSPFv3 device at the remote end of the virtual link..

```
device(config-ospf6-router)# area 1 virtual-link 10.1.1.1 authentication ipsec spi 512 esp sha1 no-
encrypt 1134567890223456789012345678901234567890
```

IPsec is configured on the specified virtual link in OSPF area 1. The device ID associated with the virtual link neighbor is 10.1.1.1, the SPI value is 512, and the Encapsulating Security Payload (ESP) protocol is selected. Secure Hash Algorithm 1 (SHA-1) authentication is enabled. The 40-character key is not encrypted in **show** command displays.

The following example configures IPsec on an OSPFv3 area.

```
device# configure terminal
device(config)# ip router-id 10.1.1.1
device(config)# ipv6 router ospf
device(config-ospf6-router)# area 1 virtual-link 10.1.1.1 authentication ipsec spi 512 esp sha1 no-encrypt
1134567890223456789012345678901234567890
```

Specifying the key rollover timer

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.

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-rollover-interval** command and specify the desired interval to set the timing of the configuration changeover.

```
device(config-ospf6-router)# key-rollover-interval 240
```

The following example sets the timing of the configuration changeover to 240 seconds (4 minutes).

```
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# ipv6 router ospf
device(config-ospf6-router)# key-rollover-interval 240
```

Clearing IPsec statistics

Statistics related to IP security (IPsec) can be cleared using the **clear ipsec statistics** command.

Use the **show ipsec statistics** command to display the IPsec statistics. After using the **clear ipsec statistics** command to clear the IPsec statistics, re-enter the **show ipsec statistics** command to verify the IPsec statistics have been cleared. The **clear ipsec statistics** command resets the IPsec packet statistics and IPsec error statistics counters to zero.

1. Enter the **exit** command as necessary to access user EXEC mode

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

2. Enter the **show ipsec statistics** command to display statistics related to IPsec.

```
device# show ipsec statistics
IPSecurity Statistics
secEspCurrentInboundSAs 1          ipsecEspTotalInboundSAs: 2
secEspCurrentOutboundSA 1          ipsecEspTotalOutboundSAs: 2
IPSecurity Packet Statistics
secEspTotalInPkts: 20              ipsecEspTotalInPktsDrop: 0
secEspTotalOutPkts: 84
IPSecurity Error Statistics
secAuthenticationErrors 0
secReplayErrors: 0                ipsecPolicyErrors: 13
secOtherReceiveErrors: 0          ipsecSendErrors: 0
secUnknownSpiErrors: 0
```

3. Enter the **clear ipsec statistics** command to clear statistics related to IPsec from the configuration.

```
device# clear ipsec statistics
```

4. Enter the **show ipsec statistics** command to verify that statistics related to IPsec have been cleared from the configuration.

```
device# show ipsec statistics
                        IPSECURITY Statistics
ipsecEspCurrentInboundSAs 0      ipsecEspTotalInboundSAs: 0
ipsecEspCurrentOutboundSA 0      ipsecEspTotalOutboundSAs: 0
                        IPSECURITY Packet Statistics
ipsecEspTotalInPkts: 0          ipsecEspTotalInPktsDrop: 0
ipsecEspTotalOutPkts: 0
                        IPSECURITY Error Statistics
ipsecAuthenticationErrors 0
ipsecReplayErrors: 0            ipsecPolicyErrors: 0
ipsecOtherReceiveErrors: 0      ipsecSendErrors: 0
ipsecUnknownSpiErrors: 0
device#
```

The counters holding IPsec packet statistics and IPsec error statistics are reset to 0.

The following example clears IPsec statistics and verifies that the IPsec statistics have been cleared.

```
device(config-ospf6-router)# exit
device(config)# exit
device# show ipsec statistics
device# clear ipsec statistics
device# show ipsec statistics
```

Displaying OSPFv3 results

The **show ipv6 ospf** command and its variations can be used to display information about OSPFv3 configurations.

Use one or more of the following commands to verify OSPFv3 information. Using the **show ipv6 ospf** command is optional, and the variations of the command can be entered in any order.

1. Enter the **exit** command as necessary to access user EXEC mode.

```
device# exit
```

2. Enter the **show ipv6 ospf** command to display general OSPFv3 information.

```
device> show ipv6 ospf

OSPFv3 Process number 0 with Router ID 0xc0a862d5(10.168.98.213)
Running 0 days 2 hours 55 minutes 36 seconds
Number of AS scoped LSAs is 4
Sum of AS scoped LSAs Checksum is 18565
External LSA Limit is 250000
Database Overflow Interval is 10
Database Overflow State is NOT OVERFLOWED
Route calculation executed 15 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: CONNECTED RIP
High Priority Message Queue Full count: 0
Graceful restart helper is enabled, strict lsa checking is disabled
Nonstop-routing is ENABLED
```

3. The following example of the **show ipv6 ospf area** command shows detailed output for assigned OSPFv3 Area 1.

```
device> show ipv6 ospf area 1

Area 1:
Authentication: Not Configured
Active interface(s) attached to this area: None
Inactive interface(s) attached to this area: ve 20 ve 30
Number of Area scoped LSAs is 311
Sum of Area LSAs Checksum is 9e8fff
Statistics of Area 1:
  SPF algorithm executed 10 times
  SPF last updated: 5920 sec ago
  Current SPF node count: 1
    Router: 1 Network: 0
  Maximum of Hop count to nodes: 0
```

4. The following example of the **show ipv6 ospf interface brief** command shows limited OSPFv3 interface information.

```
device> show ipv6 ospf interface brief

Interface Area Status Type Cost State Nbrs (F/C)
eth 1/1/1 0 up BCST 1 DROther 1/1
loopback 1 0 up BCST 1 Loopback 0/0
```

5. The following example of the **show ipv6 ospf neighbor** command shows OSPFv3 neighbor information for the device.

```
device> show ipv6 ospf neighbor

Total number of neighbors in all states: 2
Number of neighbors in state Full : 2
RouterID Pri State DR BDR Interface [State]
192.168.98.111 1 Full 192.168.98.111 192.168.98.213 e 4/3/1 [BDR]
192.168.98.111 1 Full 192.168.98.111 192.168.98.213 ve 17 [BDR]
```

6. The following example of the **show ipv6 ospf virtual-neighbor** command shows information about an OSPFv3 virtual neighbor.

```
device> show ipv6 ospf virtual-neighbor

Index Router ID Address State Interface
1 10.14.14.14 2001:db8:44:44::4 Full eth 1/1/8
Option: 00-00-00 QCount: 0 Timer: 408
2 10.14.14.14 2001:db8:44:44::4 Full tunnel 256
Option: 00-00-00 QCount: 0 Timer: 43
```

7. The following example of the **show ipv6 ospf database** command shows information about different OSPFv3 LSAs.

```
device> show ipv6 ospf database

LSA Key - Rtr:Router Net:Network Inap:InterPrefix Inar:InterRouter
Extn:ASExternal Grp:GroupMembership Typ7:Type7 Link:Link
Iap:IntraPrefix Grc:Grace

Area ID Type LSID Adv Rtr Seq(Hex) Age Cksum Len Sync
0.0.0.200 Link 897 192.168.98.213 80000007 1277 9044 64 Yes
0.0.0.200 Link 136 192.168.98.111 80000007 582 fb0b 64 Yes
0.0.0.200 Link 2049 192.168.98.213 80000006 1277 381a 64 Yes
0.0.0.200 Link 1156 192.168.98.111 80000007 582 cf38 64 Yes
0.0.0.200 Link 2052 192.168.98.213 80000004 799 5b06 64 Yes
0.0.0.200 Rtr 0 192.168.98.111 800002ea 823 cb7b 56 Yes
0.0.0.200 Rtr 0 192.168.98.213 800001c7 799 8402 56 Yes
0.0.0.200 Net 1156 192.168.98.111 80000004 823 b2d2 32 Yes
0.0.0.200 Net 136 192.168.98.111 80000008 823 aed2 32 Yes
N/A Extn 0000021d 10.223.223.223 8000000a 1319 441e 32 Yes
```

8. The following example of the **show ipv6 ospf routes** command shows output for OSPFv3 routes.

```
device> show ipv6 ospf routes

Current Route count: 309
  Intra: 304 Inter: 4 External: 1 (Type1 0/Type2 1)
  Equal-cost multi-path: 56
  OSPF Type: IA- Intra, OA - Inter, E1 - External Type1, E2 - External Type2
Destination          Cost      E2Cost    Tag      Flags    Dis
E2 ::/0              2         1         0         00000003 110
Next_Hop_Router
fe80::768e:f8ff:fe3e:1800   Outgoing_Interface Adv_Router
e 4/3/1                    192.168.98.111
fe80::768e:f8ff:fe3e:1800   ve 17             192.168.98.111
Destination          Cost      E2Cost    Tag      Flags    Dis
IA 5100::192:61:1001:0/112  3         0         0         00000007 110
Next_Hop_Router
fe80::768e:f8ff:fe3e:1800   Outgoing_Interface Adv_Router
e 4/3/1                    192.168.98.111
fe80::768e:f8ff:fe3e:1800   ve 17             192.168.98.111
Destination          Cost      E2Cost    Tag      Flags    Dis
IA 5100::192:111:2:111/128  1         0         0         00000007 110
Next_Hop_Router
fe80::768e:f8ff:fe3e:1800   Outgoing_Interface Adv_Router
e 4/3/1                    192.168.98.111
fe80::768e:f8ff:fe3e:1800   ve 17             192.168.98.111
Destination          Cost      E2Cost    Tag      Flags    Dis
IA 5100::192:111:3:111/128  1         0         0         00000007 110
Next_Hop_Router
fe80::768e:f8ff:fe3e:1800   Outgoing_Interface Adv_Router
e 4/3/1                    192.168.98.111
--More--, next page: Space, next line: Return key, quit: Control-c
```

9. 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             192.168.98.213 80000004 895 6e5e 44  Yes
  Bits: E--
  Metric: 0
  Prefix Options:
  Referenced LSType: 0
  Prefix: 5100:213:213:0:192:213:1:0/112
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 1             192.168.98.190 80001394 643 1cc9 28  Yes
  Bits: E--
  Metric: 1
  Prefix Options:
  Referenced LSType: 0
  Prefix: ::/0
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             192.168.98.71 80000258 132 a3ff 32  Yes
  Bits: E-T
  Metric: 1
  Prefix Options:
  Referenced LSType: 0
  Prefix: ::/0
  Tag: 1
```

10. The following example of the **show ipv6 ospf database** command with the **tree** shows information about the SPF trees.

```
device> show ipv6 ospf spf tree
SPF tree for Area 0
+- 10.223.223.223 cost 0
+- 10.223.223.223:88 cost 1
+- 10.1.1.1:0 cost 1
```

11. The following example of the **show ipv6 ospf database** command with the **table** shows information about the SPF table.

```
device> show ipv6 ospf spf table
SPF table for Area 0
  Destination      Bits Options  Cost  Nexthop                Interface
R 192.168.98.111   --V-B V6E---R-   1  fe80::768e:f8ff:fe3e:1800 e 4/3/1
R 192.168.98.111   --V-B V6E---R-   1  fe80::768e:f8ff:fe3e:1800 ve 17
N 192.168.98.111[136] ----- V6E---R-   1  ::                      e 4/3/1
N 192.168.98.111[1156] ----- V6E---R-   1  ::                      ve 17
```

12. The following example of the **show ipv6 ospf redistribute route** command shows information about routes that the device has redistributed into OSPFv3.

```
device> show ipv6 ospf redistribute route

Id      Prefix                                Protocol Metric Type  Metric
1       5100::192:213:163:0/112                Connect Type-2    0
2       5100:213:213:0:192:213:1:0/112          Connect Type-2    0
```

13. The following example of the **show ipv6 ospf routes** command shows information about a specified OSPFv3 route.

```
device> show ipv6 ospf routes 2001::192:111:42:111
Destination      Cost      E2Cost    Tag      Flags      Dis
IA 2001::192:111:42:111/128    1          0         0         00000007 110
  Next_Hop_Router      Outgoing_Interface Adv_Router
  fe80::768e:f8ff:fe3e:1800 e 4/3/1      10.168.98.111
  fe80::768e:f8ff:fe3e:1800 ve 17        10.168.98.111
```


BGP4

• BGP4 overview.....	298
• Implementation of BGP4.....	302
• BGP4 restart.....	303
• Basic configuration and activation for BGP4.....	306
• BGP4 parameters.....	307
• Memory considerations.....	308
• Enabling BGP4 on the device.....	309
• Changing the device ID.....	309
• Setting the local AS number.....	309
• Setting the local AS number for VRF instances.....	310
• Adding a loopback interface.....	311
• Adding BGP4 neighbors.....	311
• Disabling BGP4.....	321
• Changing the Keep Alive Time and Hold Time.....	322
• Changing the BGP4 next-hop update timer.....	322
• Enabling fast external fallover.....	323
• Changing the maximum number of paths for BGP4 Multipath load sharing.....	323
• Customizing BGP4 Multipath load sharing.....	325
• Specifying a list of networks to advertise.....	326
• Changing the default local preference.....	326
• Using the IP default route as a valid next-hop for a BGP4 route.....	327
• Changing the default MED (Metric) used for route redistribution.....	327
• Enabling next-hop recursion.....	328
• Changing administrative distances.....	330
• Requiring the first AS to be the neighbor AS.....	331
• Disabling or re-enabling comparison of the AS-Path length.....	332
• Enabling or disabling comparison of device IDs.....	332
• Configuring the device to always compare Multi-Exit Discriminators.....	333
• Treating missing MEDs as the worst MEDs.....	334
• Configuring route reflection parameters.....	334
• Configuring confederations.....	336
• Aggregating routes advertised to BGP4 neighbors.....	339
• Configuring BGP4 restart.....	340
• Modifying redistribution parameters.....	345
• Filtering.....	347
• Four-byte Autonomous System Numbers (AS4).....	363
• BGP4 AS4 attribute errors.....	368
• Configuring route flap dampening.....	368
• Generating traps for BGP4.....	372
• Configuring BGP4.....	373
• Entering and exiting the address family configuration level.....	374
• BGP route reflector.....	374
• Specifying a maximum AS path length.....	377
• BGP4 max-as error messages.....	379
• Originating the default route.....	379
• Changing the default metric used for route cost.....	379
• Configuring a static BGP4 network	380
• Generalized TTL Security Mechanism support.....	383
• Displaying BGP4 information.....	383

- Clearing traffic counters..... 419

BGP4 overview

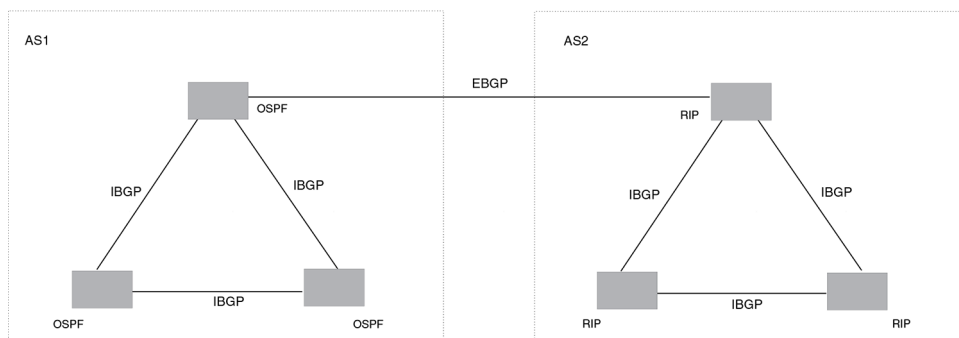
Border Gateway Protocol version 4 (BGP4) is an exterior gateway protocol that performs inter-autonomous system (AS) or inter-domain routing. It peers to other BGP-speaking systems over TCP to exchange network reachability and routing information. BGP primarily performs two types of routing: inter-AS routing, and intra-AS routing. BGP peers belonging to different autonomous systems use the inter-AS routing, referred as Exterior BGP (EBGP). On the other hand, within an AS BGP can be used to maintain a consistent view of network topology, to provide optimal routing, or to scale the network.

BGP is a path vector protocol and implements this scheme on large scales by treating each AS as a single point on the path to any given destination. For each route (destination), BGP maintains the AS path and uses this to detect and prevent loops between autonomous systems.

Devices within an AS can use different Interior Gateway Protocols (IGPs) such as RIP 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 Brocade devices.

This is a simple example of two BGP4 ASs. Each AS contains three BGP4 devices. All of the BGP4 devices within an AS communicate using iBGP. BGP4 devices communicate with other autonomous systems using eBGP. Notice that each of the devices also is running an Interior Gateway Protocol (IGP). The devices in AS1 are running OSPF and the devices in AS2 are running RIP. The device can be configured to redistribute routes among BGP4, RIP, and OSPF. They also can redistribute static routes.

FIGURE 29 BGP4 autonomous systems



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.

Relationship between the BGP4 route table and the IP route table

The device BGP4 route table can have multiple routes or paths to the same destination, which are learned from different BGP4 neighbors. A BGP4 neighbor is another device that also is running BGP4. BGP4 neighbors communicate using Transmission Control Protocol (TCP) port 179 for BGP4 communication. When you configure the device for BGP4, one of the configuration tasks you perform is to identify the device's BGP4 neighbors.

Although a device's BGP4 route table can have multiple routes to the same destination, the BGP4 protocol evaluates the routes and chooses only one to send to the IP route table. The route that BGP4 chooses and sends to the IP route table is the preferred route. This route is what the device advertises to other BGP4 neighbors. If the preferred route goes down, BGP4 updates the route information in the IP route table with a new BGP4 preferred route.

NOTE

If IP load sharing is enabled and you enable multiple equal-cost paths for BGP4, BGP4 can select more than one equal-cost path to a destination.

A BGP4 route consists of the following information:

- Network number (prefix) - A value made up of the network mask bits and an IP address; for example, 10.215.129.0/18 indicates a network mask of 18 bits applied to the IP address 10.215.129.0. When a BGP4 device advertises a route to one of its neighbors, it uses this format.
- AS-path - A list of the other autonomous systems through which a route passes. BGP4 devices can use the AS-path to detect and eliminate routing loops. For example, if a route received by a BGP4 device contains the AS that the device is in, the device does not add the route to its own BGP4 table. (The BGP4 RFCs refer to the AS-path as "AS_PATH", and RFC 4893 uses "AS4_PATH" in relation to AS4s.)
- Additional path attributes - A list of additional parameters that describe the route. The route MED and next hop are examples of these additional path attributes.

NOTE

The device re-advertises a learned best BGP4 route to the device's neighbors even when the software does not select that route for installation in the IP route table. This can happen if a route from another protocol, for example, OSPF, is preferred. The best BGP4 route is the route that BGP4 selects based on comparison of the BGP4 route path's attributes.

After a device successfully negotiates a BGP4 session with a neighbor (a BGP4 peer), the device exchanges complete BGP4 route tables with the neighbor. After this initial exchange, the device and all other RFC 1771-compliant BGP4 devices send UPDATE messages to inform neighbors of new, changed, or no longer feasible routes. BGP4 devices do not send regular updates. However, if configured to do so, a BGP4 device does regularly send KEEPALIVE messages to its peers to maintain BGP4 sessions with them if the device does not have any route information to send in an UPDATE message.

How BGP4 selects a path for a route (BGP best path selection algorithm)

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.

1. Is the next hop accessible through an Interior Gateway Protocol (IGP) route? If not, ignore the route.

NOTE

The device does not use the default route to resolve BGP4 next hop.

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 originated locally (by this BGP4 device).

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.

NOTE

This step can be skipped if **BGP4-as-path-ignore** 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.

If the routes were learned from the same neighboring AS, BGP4 compares the MEDs of two otherwise equivalent paths. This behavior is called deterministic MED. Deterministic MED is always enabled and cannot be disabled. You can also enable the device to always compare the MEDs, regardless of the AS information in the paths. To enable this comparison, enter the **always-compare-med** command at the BGP4 configuration level of the CLI. This option is disabled by default.

NOTE

By default, value 0 (most favorable) is used in MED comparison when the MED attribute is not present. The default MED comparison results in the device favoring the route paths that are missing their MEDs. You can use the **med-missing-as-worst** command to make the device regard a BGP4 route with a missing MED attribute as the least favorable path, when comparing the MEDs of the route paths.

NOTE

MED comparison is not performed for internal routes originated within the local AS or confederation unless the **compare-med-empty-aspath** command is configured.

8. Prefer routes 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 prefer the route that comes from the BGP4 device with the lowest device ID.

NOTE

Brocade devices support BGP4 load sharing among multiple equal-cost paths. BGP4 load sharing enables the device to balance traffic across the multiple paths instead of choosing just one path based on device ID. For EBGp routes, load sharing applies only when the paths are from neighbors within the same remote AS. EBGp paths from neighbors in different autonomous systems are not compared, unless multipath **multi-as** is enabled.

11. If the **compare-router ID** is enabled, prefer the path that comes from the BGP4 device with the lowest device ID. If a path contains originator ID attributes, then 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.

NOTE

When equal cost multipath (ECMP) is configured, the first established route will be the best path.

BGP4 message types

BGP4 devices communicate with neighbors (other BGP4 devices) using the following types of messages:

- OPEN
- UPDATE
- KEEPALIVE
- NOTIFICATION
- ROUTE REFRESH

OPEN message

After a BGP4 device establishes a TCP connection with a neighboring BGP4 device, the devices exchange OPEN messages. An open message indicates the following:

- BGP4 version - Indicates the version of the protocol that is in use on the device. BGP4 version 4 supports Classless Interdomain Routing (CIDR) and is the version most widely used in the Internet. Version 4 also is the only version supported on devices.
- AS number - An autonomous system number (ASN) identifies the AS to which the BGP4 device belongs. The number can be up to four bytes.
- Hold Time - The number of seconds a BGP4 device will wait for an UPDATE or KEEPALIVE message (described below) from a BGP4 neighbor before assuming that the neighbor is not operational. BGP4 devices exchange UPDATE and KEEPALIVE messages to update route information and maintain communication. If BGP4 neighbors are using different Hold Times, the lowest Hold Time is used by the neighbors. If the Hold Time expires, the BGP4 device closes the TCP connection to the neighbor and clears any information it has learned and cached from the neighbor.

You can configure the Hold Time to be 0, in which case a BGP4 device will consider neighbors to always be up. For directly-attached neighbors, you can configure the device to immediately close the TCP connection to the neighbor and clear entries learned from an EBGP neighbor if the interface to that neighbor goes down. This capability is provided by the fast external fail over feature, which is disabled by default.

- BGP4 Identifier - The device ID. The BGP4 Identifier (device ID) identifies the BGP4 device to other BGP4 devices. The device use the same device ID for OSPF and BGP4. If you do not set a device ID, the software uses the IP address on the lowest numbered loopback interface configured on the device. If the device does not have a loopback interface, the default device ID is the lowest numbered IP address configured on the device.
- Parameter list - An optional list of additional parameters used in peer negotiation with BGP4 neighbors.

UPDATE message

After BGP4 neighbors establish a BGP4 connection over TCP and exchange their BGP4 routing tables, they do not send periodic routing updates. Instead, a BGP4 neighbor sends an update to a neighbor when it has a new route to advertise or routes have changed or become unfeasible. An UPDATE message can contain the following information:

- Network Layer Reachability Information (NLRI) - The mechanism by which BGP4 supports Classless Interdomain Routing (CIDR). An NLRI entry consists of an IP prefix that indicates a network being advertised by the UPDATE message. The prefix consists of an IP network number and the length of the network portion of the number. For example, an UPDATE message with the NLRI entry 10.215.129.0/18 indicates a route to IP network 10.215.129.0 with network mask 255.255.192.0. The binary equivalent of this mask is 18 consecutive one bits, thus "18" in the NLRI entry.
- Path attributes - Parameters that indicate route-specific information such as Autonomous System path information, route preference, next hop values, and aggregation information. BGP4 uses path attributes to make filtering and routing decisions.

- Unreachable routes – A list of routes that have been in the sending device BGP4 table but are no longer feasible. The UPDATE message lists unreachable routes in the same format as new routes: *IP address* and *CIDR prefix*.

KEEPALIVE message

BGP4 devices do not regularly exchange UPDATE messages to maintain BGP4 sessions. For example, if a device configured to perform BGP4 routing has already sent the latest route information to peers in UPDATE messages, the device does not send more UPDATE messages. Instead, BGP4 devices send KEEPALIVE messages to maintain BGP4 sessions. KEEPALIVE messages are 19 bytes long and consist only of a message header. They do not contain routing data.

BGP4 devices send KEEPALIVE messages at a regular interval, called the Keep Alive Time. The default Keep Alive Time is 60 seconds.

A parameter related to the Keep Alive Time is the Hold Time. The Hold Time for a BGP4 device determines how many seconds the device waits for a KEEPALIVE or UPDATE message from a BGP4 neighbor before deciding that the neighbor is not operational. The Hold Time is negotiated when BGP4 devices exchange OPEN messages, the lower Hold Time is then used by both neighbors. For example, if BGP4 device A sends a Hold Time of 5 seconds and BGP4 device B sends a Hold Time of 4 seconds, both devices use 4 seconds as the Hold Time for their BGP4 session. The default Hold Time is 180 seconds. Generally, the Hold Time is configured to three times the value of the Keep Alive Time.

If the Hold Time is 0, a BGP4 device assumes that a neighbor is alive regardless of how many seconds pass between receipt of UPDATE or KEEPALIVE messages.

NOTIFICATION message

When you close the BGP4 session with a neighbor, the device detects an error in a message received from the neighbor, or an error occurs on the device, the device sends a NOTIFICATION message to the neighbor. No further communication takes place between the BGP4 device that sent the NOTIFICATION and the neighbors that received the NOTIFICATION.

REFRESH message

BGP4 sends a REFRESH message to a neighbor to request that the neighbor resend route updates. This type of message can be useful if an inbound route filtering policy has been changed.

Grouping of RIB-out peers

To improve efficiency in the calculation of outbound route filters, the device groups BGP4 peers together based on their outbound policies. To reduce RIB-out memory usage, the device then groups the peers within an outbound policy group according to their RIB-out routes. All peers sharing a single RIB-out route (up to 32 peers per group) also share a single physical RIB-out entry, resulting in as much as a 30-fold memory usage reduction.

NOTE

RIB-out peer grouping is not shared between different VRFs or address families.

Implementation of BGP4

BGP4 is described in RFC 1771 and the latest BGP4 drafts. The Brocade BGP4 implementation fully complies with RFC 1771. Brocade BGP4 implementation also supports the following RFCs:

- RFC 1745 (OSPF Interactions)

- RFC 1997 (BGP Communities Attributes)
- RFC 2385 (TCP MD5 Signature Option)
- RFC 2439 (Route Flap Dampening)
- RFC 2796 (Route Reflection)
- RFC 2842 (Capability Advertisement)
- RFC 3065 (BGP4 Confederations)
- RFC 2858 (Multiprotocol Extensions)
- RFC 2918 (Route Refresh Capability)
- RFC 3392 (BGP4 Capability Advertisement)
- RFC 4893 BGP Support for Four-octet AS Number Space
- RFC 3682 Generalized TTL Security Mechanism, for eBGP Session Protection

BGP4 restart

BGP4 restart is a high-availability routing feature that minimizes disruption in traffic forwarding, diminishes route flapping, and provides continuous service during a system restart, switchover, failover, or hitless OS upgrade. During such events, routes remain available between devices. BGP4 restart operates between a device and its peers, and must be configured on each participating device.

Under normal operation, when a BGP4 device is restarted, the network is automatically reconfigured. Routes available through the restarting device are deleted when the device goes down, and are then rediscovered and added back to the routing tables when the device is back up and running. In a network with devices that regularly restart, performance can degrade significantly and limit the availability of network resources. BGP4 restart dampens the network response and limits route flapping by allowing routes to remain available between devices during a restart. BGP4 restart operates between a device and peers, and must be configured on each participating device.

BGP4 restart is enabled globally by default.

A BGP4 restart-enabled device advertises the capability to establish peering relationships with other devices. When a restart begins, neighbor devices mark all of the routes from the restarting device as stale, but continue to use the routes for the length of time specified by the restart timer. After the device is restarted, it begins to receive routing updates from the peers. When it receives the end-of-RIB marker that indicates it has received all of the BGP4 route updates, it recomputes the new routes and replaces the stale routes in the route map with the newly computed routes. If the device does not come back up within the time configured for the purge timer, the stale routes are removed.

NOTE

BGP4 restart is supported in ICX switches in a stack.

The implementation of BGP4 Restart supports the following Internet Draft:

- Draft-ietf-idr-restart-10.txt: restart mechanism for BGP4

BGP4 Peer notification during a management module switchover

The BGP4 Peer notification process restores BGP4 adjacency quickly and allows packet forwarding between the newly active management module and the BGP4 peers. The handling of TCP packets with an MD5 digest prevents the silent dropping of TCP packets without triggering a RESET packet.

BGP4 restart

The BGP4 peer notification process operates effectively when implemented for the following processes that involve the intentional switching of the active status from one management module to another:

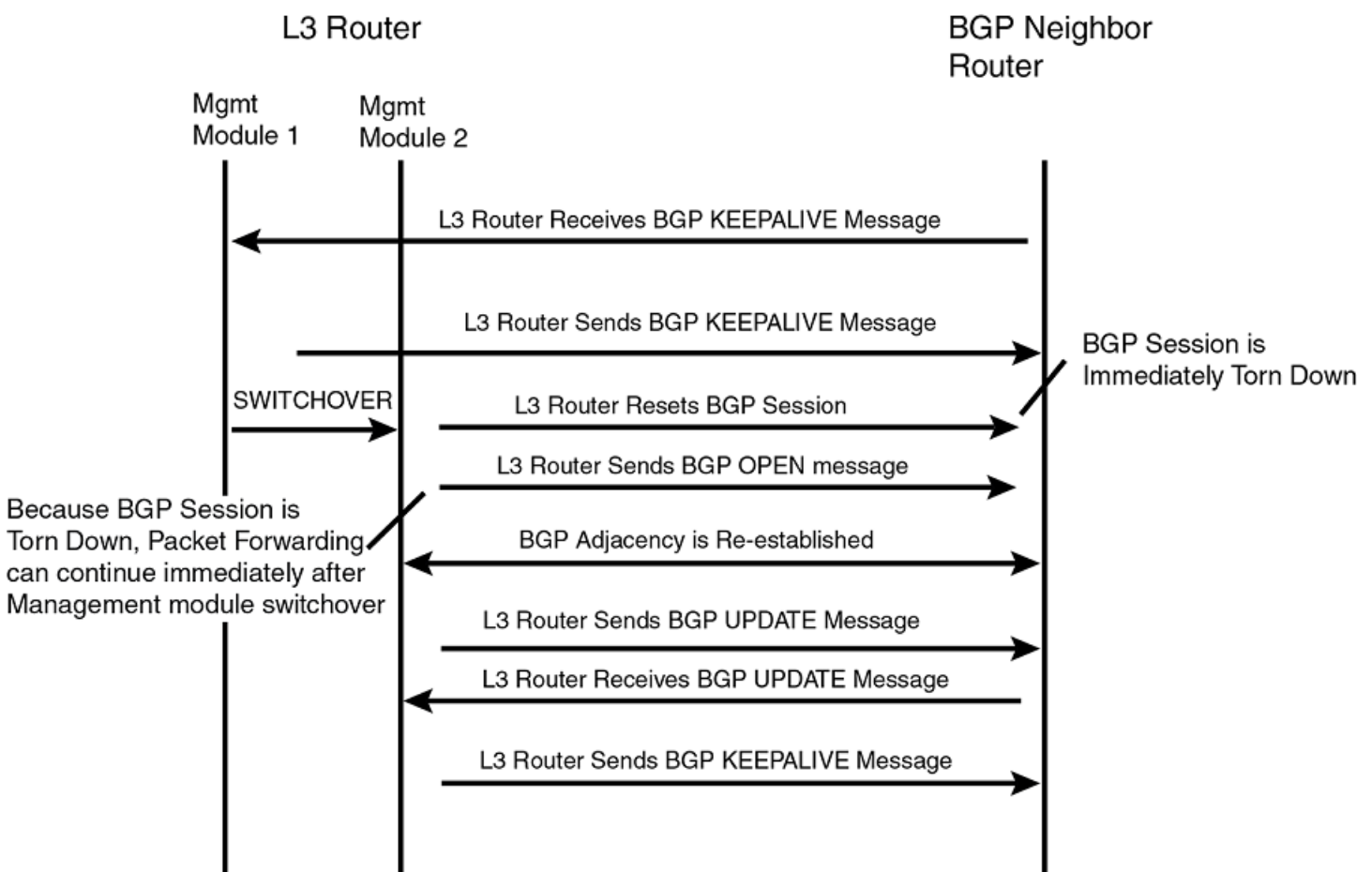
- System Reload - When a device undergoes the reload process, both management modules and all interface modules are rebooted. All BGP4 sessions are terminated BEFORE the system triggers the hardware reset.
- Switchover Requested by User - Switching over to a standby management module can be triggered by the **switchover**, **reset**, **reload**, and **hitless-reload** commands. When these commands are executed, the active management module resets the BGP4/TCP sessions with BGP4 neighbors before transferring control to the standby management module.

NOTE

Restart-enabled BGP4 sessions are not reset. The BGP4 restart protocol allows a BGP4 session to reconnect gracefully without going through the normal process.

This example describes the procedure used between the management modules in a device and a BGP4 neighbor device.

FIGURE 30 Management module switchover behavior for BGP4 peer notification



If the active management module fails due to a fault, the management module does not have the opportunity to reset BGP4 sessions with neighbors as described for intentional failovers. In this situation the management module will reboot, or the standby management module becomes the new active management module. Since the new active management module does not have the TCP/BGP4 information needed to reset the previous sessions, a remote BGP4 peer session is only reset when it sends a BGP4/TCP keep-alive packet to this device, or when the BGP4 hold-time expires.

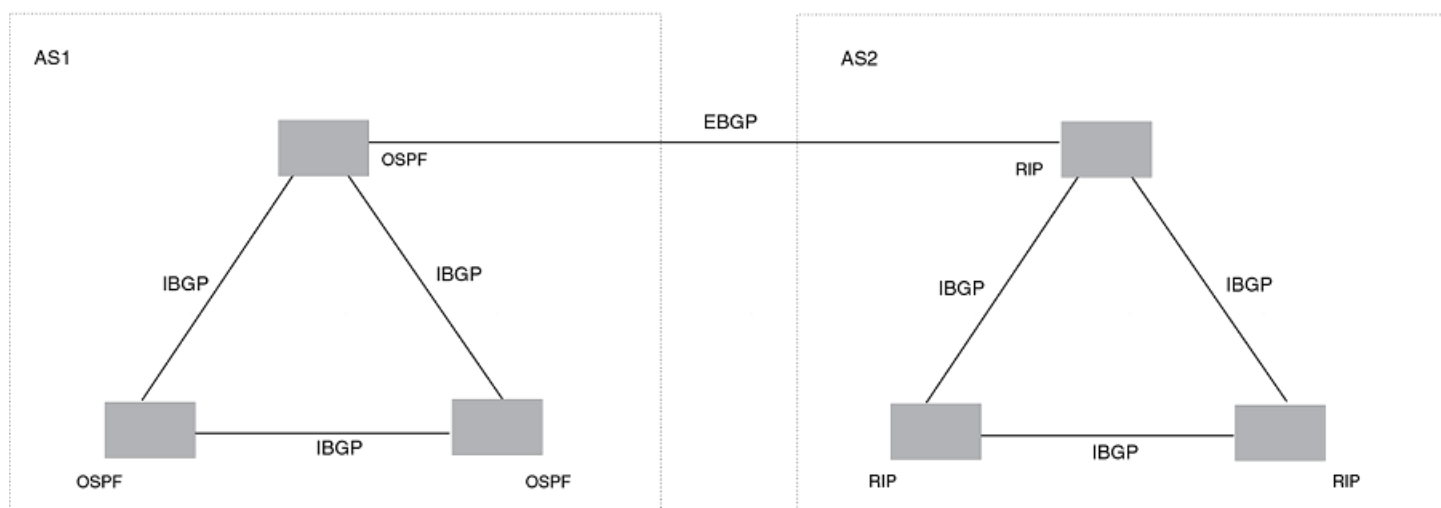
To help reduce the reconnection time after a management module failover or system reload, if an incoming TCP packet contains an MD5 digest, and no matching TCP session is found, the device attempts to find a matching BGP4 peer based on the IP address. If a BGP4 peer configuration can be found, the device looks up the MD5 password configured for the peer, and uses it to send a RESET packet.

BGP4 neighbor local AS

This feature allows you to configure a device so that it adds a peer to an AS that is different from the AS to which it actually belongs. This feature is useful when an ISP is acquired by another ISP. In this situation, customers of the acquired ISP might not want to (or might not be able to) adjust their configuration to connect to the AS of the acquiring provider.

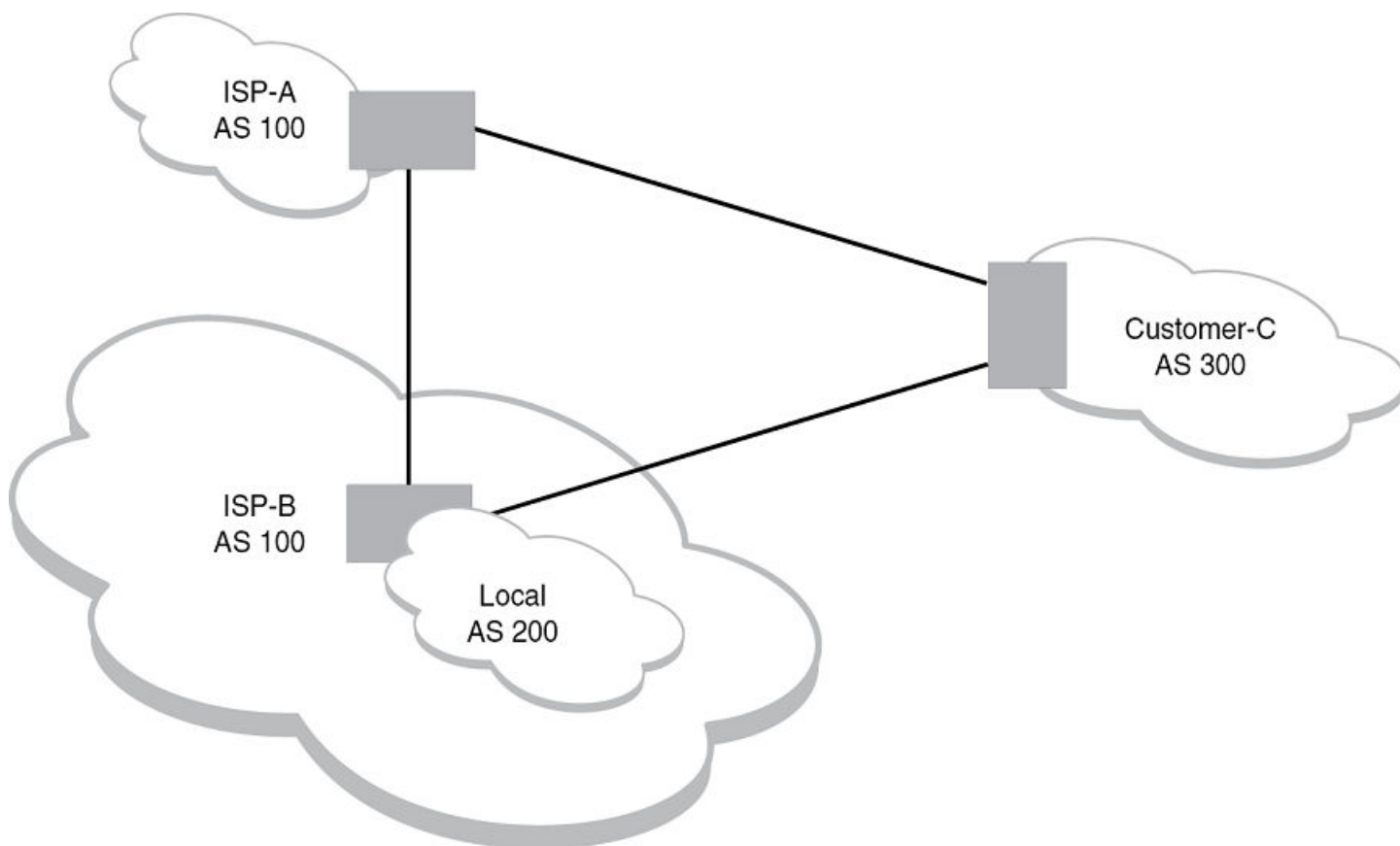
In this example, Customer C is connected to ISP-A which is in AS 100 and ISP-B which is in AS 200.

FIGURE 31 Example of customer connected to two ISPs



In the next example, ISP-A has purchased ISP-B. The AS associated with ISP-B changes to AS 100. If Customer C cannot or does not want to change their configuration or peering relationship with ISP-B, a peer with Local-AS configured with the value 200 can be established on ISP-B.

FIGURE 32 Example of Local AS configured on ISP-B



A Local AS is configured using the BGP4 **neighbor** command. To confirm that a Local AS has been configured, use the **show ip bgp neighbors** command.

Basic configuration and activation for BGP4

BGP4 is disabled by default. Follow the steps below to enable BGP4.

1. Enable the BGP4 protocol.
2. Set the local AS number.

NOTE

You must specify the local AS number for BGP4 to become functional.

3. Add each BGP4 neighbor (peer BGP4 device) and identify the AS the neighbor is in.

4. Save the BGP4 configuration information to the system configuration file.

For example, enter commands such as the following.

```
device> enable
device# configure terminal
device(config)# router bgp
BGP4: Please configure 'local-as' parameter in order to enable BGP4.
device(config-bgp)# local-as 10
device(config-bgp-router)#neighbor 10.157.23.99 remote-as 100
device(config-bgp)# write memory
```

Syntax: router bgp

The **router bgp** command enables the BGP4 protocol.

NOTE

By default, the Brocade device ID is the IP address configured on the lowest numbered loopback interface. If the device does not have a loopback interface, the default device ID is the lowest numbered IP interface address configured on the device. If you change the device ID, all current BGP4 sessions, OSPF adjacencies, and OSPFv3 adjacencies are cleared.

NOTE

When BGP4 is enabled on a Brocade device, you do not need to reset the system. The protocol is activated as soon as you enable it. The device begins a BGP4 session with a BGP4 neighbor when you add 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.

Memory considerations

- 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

The following parameter changes take effect only after the BGP4 sessions on the device are cleared, or reset using the "soft" clear option:

- Change the Hold Time or Keep Alive Time.
- Aggregate routes
- Add, change, or negate filter tables that affect inbound and outbound route policies.
- Apply maximum AS path limit settings to the RIB.

The following parameter change takes effect only after you disable and then re-enable redistribution:

- Change the default MED (metric).

Memory considerations

BGP4 can handle a very large number of routes and therefore requires a lot of memory. For example, in a typical configuration with a single BGP4 neighbor, receiving a full internet route table, a BGP4 device may need to hold over a million routes. Many configurations, especially those involving more than one neighbor, can require the device to hold even more routes. Brocade devices provide dynamic memory allocation for BGP4 data. BGP4 devices automatically allocate memory when needed to support BGP4 neighbors, routes and route attribute entries. Dynamic memory allocation is performed automatically by the software and does not require a reload.

The routes sent to and received from neighbors use the most BGP4 memory. Generally, the actual limit to the number of neighbors, routes, or route attribute entries the device can accommodate depends on how many routes the device sends to and receives from the neighbors.

In some cases, where most of the neighbors do not send or receive a full BGP route table (about 80,000 routes), the memory can support a larger number of BGP4 neighbors. However, if most of the BGP4 neighbors send or receive full BGP route tables, the number of BGP neighbors the memory can support is less than in configurations where the neighbors send smaller route tables.

Memory configuration options obsoleted by dynamic memory

Devices that support dynamic BGP4 memory allocation do not require or even support static configuration of memory for BGP4 neighbors, routes, or route attributes. Consequently, the following CLI commands and equivalent Web management options are not supported on these devices:

- **max-neighbors** *num*
- **max-routes** *num*
- **max-attribute-entries** *num*

If you boot a device that has a startup-config file that contains these commands, the software ignores the commands and uses dynamic memory allocation for BGP4. The first time you save the device running configuration (running-config) to the startup-config file, the commands are removed from the file.

Enabling BGP4 on the device

When you enable BGP4 on the device, BGP4 is automatically activated. To enable BGP4 on the device, enter the following commands.

```
device# configure terminal
device(config)# router bgp
BGP4: Please configure 'local-as' parameter in order to enable BGP4.
device(config-bgp-router)# local-as 10
device(config-bgp-router)# neighbor 10.157.23.99 remote-as 100
device(config-bgp-router)# write memory
```

Changing the device ID

The OSPF and BGP4 protocols use device IDs to identify devices that are running the protocols. A device ID is a valid, unique IP address and sometimes is an IP address configured on the device. The device ID cannot be an IP address in use by another device.

By default, the device ID on a Brocade device is one of the following:

- If the device has loopback interfaces, the default device ID is the IP address on the lowest numbered loopback interface configured on the Brocade device. For example, if you configure loopback interfaces 1, 2, and 3 as follows, the default device ID is 10.9.9.9/24:
 - Loopback interface 1, 10.9.9.9/24
 - Loopback interface 2, 10.4.4.4/24
 - Loopback interface 3, 10.1.1.1/24
- If the device does not have any loopback interfaces, the default device ID is the lowest numbered IP interface address configured on the device.

NOTE

Brocade devices use the same device ID for both OSPF and BGP4. If the device is already configured for OSPF, you may want to use the device ID already assigned to the device rather than set a new one. To display the current device ID, enter the **show ip** command at any CLI level.

To change the device ID, enter a command such as the following.

```
device(config)# ip router-id 10.157.22.26
```

Syntax: [no] ip router-id ip-addr

The *ip-addr* can be any valid, unique IP address.

NOTE

You can specify an IP address used for an interface on the Brocade device, but do not specify an IP address that is being used by another device.

Setting the local AS number

The local autonomous system number (ASN) identifies the AS in which the Brocade BGP4 device resides.

Setting the local AS number for VRF instances

To set the local AS number, enter commands such as the following.

```
device(config)# router bgp
BGP4: Please configure 'local-as' parameter in order to enable BGP4.
device(config-bgp)# local-as 10
device(config-bgp)# write memory
```

Syntax: [no] local-as *num*

The *num* parameter specifies a local AS number in the range 1 through 4294967295. It has no default. AS numbers 64512 - 65535 are the well-known private BGP4 AS numbers and are not advertised to the Internet community.

Setting the local AS number for VRF instances

The local autonomous system (AS) number identifies the AS in which the BGP4 device resides.

You can assign different BGP AS numbers for each VRF instance. If you do not assign an AS number, the BGP VRF instances use the default BGP AS number, as in previous releases.

The **local-as** command is available under the "global BGP" CLI level and "address- family ipv4 unicast vrf" CLI level.

To set the local as number for a VRF, enter commands such as the following.

```
device(config-bgp)# address-family ipv4 unicast vrf vrf-name
device(config-bgp)# local-as num
```

Syntax: [no] local-as *num*

The *num* parameter specifies a local AS number in the range 1 - 4294967295. It has no default. AS numbers 64512 - 65535 are the well-known private BGP4 AS numbers and are not advertised to the Internet community.

The configuration takes effect immediately and the BGP VRF instance is reset. All BGP peering within the VRF is reset, and take the new AS number.

The local AS number for the VRF instance, if configured, is displayed in the **show running-config** and **show ip bgp config** command output.

Enter the **show ip bgp config** command:

```
device# show ip bgp config
Current BGP configuration:
router bgp
  local-as 100
  neighbor 10.10.10.10 remote-as 200

  address-family ipv4 unicast
  exit-address-family

  address-family ipv6 unicast
  exit-address-family

  address-family ipv4 unicast vrf vrf_a
  local-as 300
  neighbor 10.111.111.111 remote-as 400
  exit-address-family
```

Adding a loopback interface

You can configure the device to use a loopback interface instead of a specific port or virtual routing interface to communicate with a BGP4 neighbor. A loopback interface adds stability to the network by working around route flap problems that can occur due to unstable links between the device and neighbors.

Loopback interfaces are always up, regardless of the states of physical interfaces. Loopback interfaces are especially useful for IBGP neighbors (neighbors in the same AS) that are multiple hops away from the device. When you configure a BGP4 neighbor on the device, you can specify whether the device uses the loopback interface to communicate with the neighbor. As long as a path exists between the device and the neighbor, BGP4 information can be exchanged. The BGP4 session is not associated with a specific link, but is instead associated with the virtual interfaces.

NOTE

If you configure the Brocade device to use a loopback interface to communicate with a BGP4 neighbor, the peer IP address on the remote device pointing to your loopback address must be configured.

To add a loopback interface, enter commands such as the following.

```
device(config-bgp) # exit
device(config) # int loopback 1
device(config-lbif-1) # ip address 10.0.0.1/24
```

Syntax: `[no] interface loopback num`

The *num* value can be from 1 through the maximum number of loopback interfaces supported on the device.

Adding BGP4 neighbors

Because BGP4 does not contain a peer discovery process, for each BGP4 neighbor (peer), you must indicate the IP address and the AS number of each neighbor. Neighbors that are in different autonomous systems communicate using EBGP. Neighbors within the same AS communicate using IBGP.

NOTE

If the device has multiple neighbors with similar attributes, you can simplify configuration by configuring a peer group, then adding individual neighbors to it. The configuration steps are similar, except you specify a peer group name instead of a neighbor IP address when configuring the neighbor parameters, then add individual neighbors to the peer group.

NOTE

The device attempts to establish a BGP4 session with a neighbor as soon as you enter a command specifying the IP address of the neighbor. If you want to completely configure the neighbor parameters before the device establishes a session with the neighbor, you can administratively shut down the neighbor.

To add a BGP4 neighbor with an IP address 10.157.22.26, enter the following command.

```
device(config-bgp-router) # neighbor 10.157.22.26 remote-as 100
```

The neighbor *ip-addr* must be a valid IP address.

The **neighbor** command has additional parameters, as shown in the following syntax:

Syntax: `no neighbor [ip-addr | peer-group-name] [{activate}] [advertisement-interval seconds] [allowas-in num] [capability as4 [enable | disable]] [capability orf prefixlist [send | receive]] [default-originate [route-map map-name]] [description string] [distribute-list in | out num,num,... | ACL-num localin | out] [ebgp-btsh] [ebgp-multihop [num]] [enforce-first-as] [filter-list access-list-name [in | out]] [local-as as-num [no-prepend]] [maxas-limit in [num | disable]] [maximum-prefix num [threshold]] [teardown] [next-hop-self] [password]`

string] [**peer-group** *group-name*] [**prefix-list** *string* **in** | **out**] [**remote-as** *as-number*] [**remove-private-as**] [**route-map** **in** | **out** *map-name*] [**route-reflector-client**] [**send-community**] [**shutdown** [**generate-rib-out**]] [**soft-reconfiguration** **inbound**] [**timers** **keep-alive** *num* **hold-time** *num*] [**unsuppress-map** *map-name*] [**update-source** *ip-addr* | **ethernet** *unit / slot / portnum* | **loopback** *num* | **ve** *num*] [**weight** *num*] [**send-label**]

The *ip-addr* and *peer-group-name* parameters indicate whether you are configuring an individual neighbor or a peer group. If you specify a neighbor IP address, you are configuring that individual neighbor. If you specify a peer group name, you are configuring a peer group.

activate allows exchange of routes in the current family mode.

advertisement-interval *seconds* configures an interval in seconds over which the specified neighbor or peer group will hold all route updates before sending them. At the expiration of the timer, the routes are sent as a batch. The default value for this parameter is zero. Acceptable values are 0 to 3600 seconds.

NOTE

The device applies the advertisement interval only under certain conditions. The device does not apply the advertisement interval when sending initial updates to a BGP4 neighbor. As a result, the device sends the updates one immediately after another, without waiting for the advertisement interval.

allows-in *num* disables the AS_PATH check function for routes learned from a specified location. BGP4 usually rejects routes that contain an AS number within an AS_PATH attribute to prevent routing loops.

capability as4 [**enable** | **disable**] enables the capability of processing AS4s. The optional keywords **enable** and **disable** specify whether the feature should be changed from its current state. For example, if this neighbor belongs to a peer group that is enabled for AS4s but you want to disable it on the current interface, use the command and include the **disable** keyword.

capability orf prefixlist [**send** | **receive**] configures cooperative device filtering. The **send** and **receive** parameters specify the support you are enabling:

- **send** - The device sends the IP prefix lists as Outbound Route Filters (ORFs) to the neighbor.
- **receive** - The device accepts filters as Outbound Route Filters (ORFs) from the neighbor.

If you do not specify either **send** or **receive**, both capabilities are enabled. The **prefixlist** parameter specifies the type of filter you want to send to the neighbor.

NOTE

The current release supports cooperative filtering only for filters configured using IP prefix lists.

default-originate [**route-map** *map-name*] configures the device to send the default route 0.0.0.0 to the neighbor. If you use the *route-map map-name* parameter, the route map injects the default route conditionally, based on the following supported match conditions in the route map.

- BGP match IP
- BGP match IP address prefix list
- BGP match IP address access list
- BGP match IPV6 address prefix list

description *string* specifies a name for the neighbor. You can enter an alphanumeric text string up to 80 characters long.

distribute-list in | **out** *num,num,...* specifies a distribute list to be applied to updates to or from the specified neighbor. The **in** and **out** keywords specify whether the list is applied on updates received from the neighbor, or sent to the neighbor. The *num,num,...* parameter specifies the list of address-list filters. The device applies the filters in the order in which you list them and stops applying the filters in the distribute list when a match is found.

To use an IP ACL instead of a distribute list, you can specify **distribute-list***ACL-num* **in** | **out** . In this case, *ACL-num* is an IP ACL.

NOTE

By default, if a route does not match any of the filters, the device denies the route. To change the default behavior, configure the last filter as **permit any any** .

NOTE

The address filter must already be configured.

ebgp-btsh enables GTSM protection for the specified neighbor.

ebgp-multihop [*num*] specifies that the neighbor is more than one hop away and that the session type with the neighbor is EBGp-multihop. This option is disabled by default. The *num* parameter specifies the TTL you are adding for the neighbor. You can specify a number from 0 through 255. The default is 0. If you leave the EBGp TTL value set to 0, the software uses the IP TTL value.

enforce-first-as ensures, for this neighbor, that the first AS listed in the AS_SEQUENCE field of an AS path update message from EBGp neighbors is the AS of the neighbor that sent the update.

filter-list in | **out** *num,num,...* specifies an AS-path filter list or a list of AS-path ACLs. The **in** and **out** keywords specify whether the list is applied on updates received from the neighbor or sent to the neighbor. If you specify **in** or **out**, the *num,num,...* parameter specifies the list of AS-path filters. The device applies the filters in the order in which you list them and stops applying the filters in the AS-path filter list when a match is found.

weight *num* specifies a weight that the device applies to routes received from the neighbor. You can specify a number from 0 through 65535.

Alternatively, you can specify **filter-list** *acl-num* **in** | **out** | **weight** to use an AS-path ACL instead of an AS-path filter list. In this case, *acl-num* is an AS-path ACL.

NOTE

By default, if an AS-path does not match any of the filters or ACLs, the device denies the route. To change the default behavior, configure the last filter or ACL as **permit any any** .

NOTE

The AS-path filter or ACL must already be configured.

local-as *as-num* assigns a local AS number with the value specified by the *as-num* variable to the neighbor being configured. The *as-num* has no default value. Its range is 1 - 4294967295.

NOTE

When the **local-as** option is used, the device automatically prepends the local AS number to the routes that are received from the EBGp peer; to disable this behavior, include the **no-prepend** keyword.

maxas-limit in *num* | **disable** specifies that the device discard routes that exceed a maximum AS path length received in UPDATE messages. You can specify a value from 0 - 300. The default value is 300. The **disable** keyword is used to stop a neighbor from inheriting the configuration from the peer-group or global and to the use system default value.

maximum-prefix *num* specifies the maximum number of IP network prefixes (routes) that can be learned from the specified neighbor or peer group . You can specify a value from 0 through 4294967295. The default is 0 (unlimited).

- The *num* parameter specifies the maximum number. The range is 0 through 4294967295. The default is 0 (unlimited).
- The *threshold* parameter specifies the percentage of the value you specified for the **maximum-prefix** *num* , at which you want the software to generate a Syslog message. You can specify a value from 1 (one percent) to 100 (100 percent). The default is 100.

- The **teardown** parameter tears down the neighbor session if the maximum-prefix limit is exceeded. The session remains shutdown until you clear the prefixes using the **clear ip bgp neighbor all** or **clear ip bgp neighbor** command, or change the maximum prefix configuration for the neighbor. The software also generates a Syslog message.

next-hop-self specifies that the device should list itself as the next hop in updates sent to the specified neighbor. This option is disabled by default.

password *string* specifies an MD5 password for securing sessions between the device and the neighbor. You can enter a string up to 80 characters long. The string can contain any alphanumeric characters and spaces if the words in the password are placed inside quotes.

NOTE

If you want the software to assume that the value you enter is the clear-text form, and to encrypt display of that form, do not enter 0 or 1. Instead, omit the encryption option and allow the software to use the default behavior. If you specify encryption option 1, the software assumes that you are entering the encrypted form of the password or authentication string. In this case, the software decrypts the password or string you enter before using the value for authentication. If you accidentally enter option 1 followed by the clear-text version of the password or string, authentication will fail because the value used by the software will not match the value you intended to use.

The system creates an MD5 hash of the password and uses it for securing sessions between the device and its neighbors. To display the configuration, the system uses a 2-way encoding scheme to be able to retrieve the original password that was entered.

By default, the password is encrypted. If you want the password to appear in clear text, insert a 0 between the password and the string.

```
device(config-bgp)# neighbor 10.157.22.26 password 0 marmalade
```

The system adds an encryption code followed by the encrypted text of the original password. For example, the following portion of the code has the encrypted code "2".

```
password 2 $IUA2Pwc9LW9VIW9zVQ=="
```

One of the following may be displayed:

- 0 = the password is not encrypted and is in clear text
- 2 = the password uses proprietary base64 cryptographic 2-way algorithm

peer-group *group-name* assigns the neighbor to the specified peer group.

prefix-list *string* **in** | **out** specifies an IP prefix list. You can use IP prefix lists to control routes to and from the neighbor. IP prefix lists are an alternative method to AS-path filters. The **in** and **out** keywords specify whether the list is applied on updates received from the neighbor or sent to the neighbor. The filters can use the same prefix list or different prefix lists.

You must specify a prefix-list that matches an existing prefix-list entry. An implicit deny is applied to traffic that does not match any prefix-list entry. Use the **show ip prefix-list** command to view information about configured prefix-lists.

remote-as *as-number* specifies the AS in which the remote neighbor resides. The *as-number* has no default value. The range is 1 - 4294967295.

remove-private-as configures the device to remove private AS numbers from update messages the device sends to this neighbor. The device will remove AS numbers 64512 through 65535 (the well-known BGP4 private AS numbers) from the AS-path attribute in update messages the device sends to the neighbor. This option is disabled by default.

route-map **in** | **out** *map-name* specifies a route map the device will apply to updates sent to or received from the specified neighbor. The **in** and **out** keywords specify whether the list is applied on updates received from the neighbor or sent to the neighbor.

NOTE

The route map must already be configured.

route-reflector-client specifies that this neighbor is a route-reflector client of the device. Use the parameter only if this device is going to be a route reflector. This option is disabled by default.

send-community enables sending the community attribute in updates to the specified neighbor. By default, the device does not send the community attribute.

shutdown administratively shuts down the session with this neighbor. Shutting down the session lets you configure the neighbor and save the configuration without actually establishing a session with the neighbor.

When a peer is put into the shutdown state, ribout routes are not produced for that peer. You can elect to produce ribout routes using the **generate-rib-out** option. This option is disabled by default.

soft-reconfiguration inbound enables the soft reconfiguration feature, which stores all the route updates received from the neighbor. If you request a soft reset of inbound routes, the software performs the reset by comparing the policies against the stored route updates, instead of requesting the neighbor BGP4 route table or resetting the session with the neighbor.

timers keep-alive num hold-time num overrides the global settings for the Keep Alive Time and Hold Time. For the Keep Alive Time, you can specify 0 - 65535 seconds. For the Hold Time, you can specify 0 or a number in the range 3 through 65535 (1 and 2 are not allowed). If you set the Hold Time to 0, the device waits indefinitely for messages from a neighbor without concluding that the neighbor is non-operational. The defaults for these parameters are the currently configured global Keep Alive Time and Hold Time.

unsuppress-map map-name removes route suppression from neighbor routes when those routes have been dampened due to aggregation.

update-source ip-addr | ethernetunit/slot/portnum | loopbacknum | venum configures the device to communicate with the neighbor through the specified interface. There is no default.

weight num specifies a weight a device will add to routes received from the specified neighbor. BGP4 prefers larger weights over smaller weights. The default weight is 0.

The **send-label** keyword enables IPv6 label capability for the IPv4 peers.

Removing route dampening from suppressed routes

You can selectively un-suppress specific routes that have been suppressed due to aggregation, and allow these routes to be advertised to a specific neighbor or peer group.

```
device(config-bgp)# aggregate-address 10.1.0.0 255.255.0.0 summary-only
device(config-bgp)# show ip bgp route 10.1.0.0/16 longer
Number of BGP Routes matching display condition : 2
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH S:SUPPRESSED F:FILTERED
Prefix      Next Hop      Metric      LocPrf      Weight Status
1 10.1.0.0/16 0.0.0.0      101         32768 BAL
AS_PATH:
2 10.1.44.0/24 10.2.0.1      1          101         32768 BLS
AS_PATH:
```

In this example, the **aggregate-address** command configures an aggregate address of 10.1.0.0 255.255.0.0. and the **summary-only** parameter prevents the device from advertising more specific routes contained within the aggregate route.

Entering a **show ip bgp route** command for the aggregate address 10.1.0.0/16 shows that the more specific routes aggregated into 10.1.0.0/16 have been suppressed. In this case, the route to 10.1.44.0/24 has been suppressed. If you enter this command, the display shows that the route is not being advertised to the BGP4 neighbors.

```
device(config-bgp)# show ip bgp route 10.1.44.0/24
Number of BGP Routes matching display condition : 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 S:SUPPRESSED F:FILTERED
Prefix      Next Hop      Metric      LocPrf      Weight Status
```

Adding BGP4 neighbors

```

1      10.1.44.0/24      10.2.0.1      1      101      32768  BLS
AS_PATH:
Route is not advertised to any peers

```

To override the **summary-only** parameter and allow a specific route to be advertised to a neighbor, enter commands such as the following

```

device(config)# ip prefix-list Unsuppress1 permit 10.1.44.0/24
device(config)# route-map RouteMap1 permit 1
device(config-routemap RouteMap1)# exit
device(config)# router bgp
device(config-bgp)# neighbor 10.1.0.2 unsuppress-map RouteMap1
device(config-bgp)# clear ip bgp neighbor 10.1.0.2 soft-out

```

The **ip prefix-list** command configures an IP prefix list for network 10.1.44.0/24, which is the route you want to unsuppress. The next two commands configure a route map that uses the prefix list as input. The **neighbor** command enables the device to advertise the routes specified in the route map to neighbor 10.1.0.2. The **clear** command performs a soft reset of the session with the neighbor so that the device can advertise the unsuppressed route.

Syntax: **[no] neighbor** { *ip-addr* | *peer-group-name* } **unsuppress-map** *map-name*

The **show ip bgp route** command verifies that the route has been unsuppressed.

```

device(config-bgp)# show ip bgp route 10.1.44.0/24
Number of BGP Routes matching display condition : 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 S:SUPPRESSED F:FILTERED
      Prefix      Next Hop      MED LocPrf      Weight Status
1      10.1.44.0/24      10.2.0.1      1      101      32768  BLS
AS_PATH:
Route is advertised to 1 peers:
10.1.0.2(4)

```

Encrypting BGP4 MD5 authentication keys

When you configure a BGP4 neighbor or neighbor peer group, you can specify an MD5 authentication string to authenticate packets exchanged with the neighbor or peer group of neighbors.

For added security, by default, the software encrypts the display of the authentication string. The software also provides an optional parameter to disable encryption of the authentication string, on an individual neighbor or peer group basis. By default, MD5 authentication strings are displayed in encrypted format in the output of the following commands:

- **show running-config** (or **write terminal**)
- **show configuration**
- **show ip bgp config**

When encryption of the authentication string is enabled, the string is encrypted in the CLI regardless of the access level you are using.

When you save the configuration to the startup configuration file, the file contains the new BGP4 command syntax and encrypted passwords or strings.

NOTE

Brocade recommends that you save a copy of the startup configuration file for each device you plan to upgrade.

Encryption example

The following commands configure a BGP4 neighbor and a peer group, and specify MD5 authentication strings (passwords) to authenticate packets exchanged with the neighbor or peer group.

```
device(config-bgp)# local-as 2
device(config-bgp)# neighbor xyz peer-group
device(config-bgp)# neighbor xyz password abc
device(config-bgp)# neighbor 10.10.200.102 peer-group xyz
device(config-bgp)# neighbor 10.10.200.102 password test
```

The BGP4 configuration commands appear in the following format as a result of the **show ip bgp configuration** command.

```
device# show ip bgp configuration
Current BGP configuration:
router bgp
  local-as 2
  neighbor xyz peer-group
  neighbor xyz password $b24tbw==
  neighbor 10.10.200.102 peer-group xyz
  neighbor 10.10.200.102 remote-as 1
  neighbor 10.10.200.102 password $on-o
```

In this output, the software has converted the commands that specify an authentication string into the new syntax (described below), and has encrypted display of the authentication strings.

Since the default behavior does not affect the BGP4 configuration itself but does encrypt display of the authentication string, the CLI does not list the encryption options.

Syntax: **[no] neighbor** { *ip-addr* | *peer-group-name* } **password** *string*

The *ip-addr* | *peer-group-name* parameter indicates whether you are configuring an individual neighbor or a peer group. If you specify the IP address of a neighbor, you are configuring that individual neighbor. If you specify a peer group name, you are configuring a peer group.

If you want the software to assume that the value you enter is the clear-text form and to encrypt the display of that form, do not enter 0 or 1. Instead, omit the encryption option and allow the software to use the default behavior. If you specify encryption option 1, the software assumes that you are entering the encrypted form of the password or authentication string. In this case, the software decrypts the password or string you enter before using the value for authentication. If you accidentally enter option 1 followed by the clear-text version of the password or string, authentication will fail because the value used by the software will not match the value you intended to use.

The **password** *string* parameter specifies an MD5 authentication string to secure sessions between the device and the neighbor. You can enter a string of up to 80 characters. The string can contain any alphanumeric characters, but must be placed inside quotes if it contains a space.

The system creates an MD5 hash of the password and uses it to secure sessions between the device and the neighbors. To display the configuration, the system uses a 2-way encoding scheme to retrieve the original password.

By default, password is encrypted. If you want the password to be in clear text, insert a 0 between **password** and *string*.

```
device(config-bgp)# neighbor 10.157.22.26 password admin
```

Displaying the authentication string

To display the authentication string, enter the following commands.

```
device(config)# enable password-display
device(config)# show ip bgp neighbors
```

The **enable password-display** command enables display of the authentication string, but only in the output of the **show ip bgp neighbors** command. String display is still encrypted in the startup configuration file and running configuration. Enter the command at the global CONFIG level of the CLI.

NOTE

The command also displays SNMP community strings in clear text, in the output of the **show snmp server** command.

Displaying neighbor information

To display IPv6 unicast route summary information, enter the **show ip bgp ipv6 summary** command:

```
device(config-bgp)# show ip bgp ipv6 summary
BGP4 Summary
Router ID: 10.1.1.1 Local AS Number: 1
Confederation Identifier: not configured
Confederation Peers:
Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
Number of Neighbors Configured: 1, UP: 1
Number of Routes Installed: 1, Uses 86 bytes
Number of Routes Advertising to All Neighbors: 0 (0 entries)
Number of Attribute Entries Installed: 1, Uses 90 bytes
Neighbor Address AS# State Time Rt:Accepted Filtered Sent ToSend
192.168.1.2 2 ESTAB 0h 1m51s 1 0 0 0
```

Syntax: show ip bgp ipv6 summary

To display IPv6 unicast device information with respect to the IPv4 neighbor, enter the **show ip bgp ipv6 neighbors** command:

```
device(config-bgp)# show ip bgp ipv6 neighbors
Total number of BGP Neighbors: 1
1 IP Address: 192.168.1.2, AS: 2 (EBGP), RouterID: 10.1.1.2, VRF: default-vrf
State: ESTABLISHED, Time: 0h8m33s, KeepAliveTime: 60, HoldTime: 180
KeepAliveTimer Expire in 17 seconds, HoldTimer Expire in 135 seconds
UpdateSource: Loopback 1
RefreshCapability: Received
.....
Neighbor NLRI Negotiation:
Peer Negotiated IPV6 unicast capability
Peer configured for IPV6 unicast Routes
Neighbor AS4 Capability Negotiation:
TCP Connection state: ESTABLISHED, flags:00000033 (0,0)
```

Syntax: show ip bgp ipv6 neighbors [last-packet-with-error] [routes-summary] [ip-address]

The **neighbors** parameter provides details on TCP and BGP neighbor connections. The **last-packet-with-error** parameter displays the last packet received with error. The **routes-summary** parameter displays the routes summary.

The *ip-address* parameter is the neighbor IP address. The following sub-parameters are available for the *ip-address* parameter:

[advertised routes] [flap-statistics] [last-packet-with-error] [received] [received-routes] [rib-out-routes] [routes][routes-summary]

The **advertised-routes** parameter displays routes advertised to a neighbor. The **flap-statistics** parameter displays flap statistics for a neighbor. The **last-packet-with-error** parameter displays the last packet received with error. The **received** parameter displays the received ORF from neighbor. The **received-routes** parameter displays the received routes from neighbor. The **rib-out-routes** parameter displays RIB-out routes for a neighbor. The **routes** parameter displays routes learned from neighbor. The **routes-summary** parameter displays routes summary for a neighbor.

Clearing IPv6 route information

To clear IPv6 unicast route information with respect to IPv4 neighbors, enter the **clear ip bgp ipv6 neighbor** command.

Syntax: `clear ip bgp ipv6 [neighbor] [as-number | ipaddress | peer-group-name | all]`

The **dampening** parameter clears route flap dampening information. The **flap-statistics** parameter clears route flap statistics.

The **local** parameter clears local information. The **routes** parameter clears BGP routes. The **traffic** parameter clears BGP traffic counters.

The **ipv6** parameter clears information for ipv6 address family. The **vpn4** parameter clears information for VPN4 address family. The

vrf parameter clears information for a VRF instance.

The **neighbor** parameter has the following sub-parameters:

as-number identifies neighbors with the specified AS number, 1-4294967295. *ipaddress* identifies the neighbor IP address. *peer-group-name* clears the peer group name identified using ASCII string. *all* clears all BGP neighbors.

Adding a BGP4 peer group

A peer group is a set of BGP4 neighbors that share common parameters. 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

Peer group parameters

You can set all neighbor parameters in a peer group. When you add a neighbor to the peer group, the neighbor receives all the parameter settings you set in the group, except parameter values you have explicitly configured for the neighbor. If you do not set a neighbor parameter in the peer group and the parameter also is not set for the individual neighbor, the neighbor uses the default value.

Peer group configuration rules

The following rules apply to peer group configuration:

- You must configure a peer group before you can add neighbors to the peer group.
- If you remove a parameter from a peer group, the value for that parameter is reset to the default for all the neighbors within the peer group, unless you have explicitly set that parameter on individual neighbors. In this case, the value you set on the individual neighbors applies to those neighbors, while the default value applies to neighbors for which you have not explicitly set the value.

NOTE

If you enter a command to remove the remote AS parameter from a peer group, the software makes sure that the peer group does not contain any neighbors. If the peer group contains neighbors, the software does not allow you to remove the remote AS so that the neighbors in the peer group that are using the remote AS do not lose connectivity to the device.

You can override neighbor parameters that do not affect outbound policy on an individual neighbor basis:

- If you do not specify a parameter for an individual neighbor, the neighbor uses the value in the peer group.
- If you set the parameter for the individual neighbor, that value overrides the value you set in the peer group.

- If you add a parameter to a peer group that already contains neighbors, the parameter value is applied to neighbors that do not already have the parameter explicitly set. If a neighbor has the parameter explicitly set, the explicitly set value overrides the value you set for the peer group.
- If you remove the setting for a parameter from a peer group, the value for that parameter changes to the default value for all the neighbors in the peer group that do not have that parameter individually set.

Configuring a peer group

To configure a peer group, enter commands such as the following at the BGP4 configuration level.

```
device(config-bgp-router)# neighbor PeerGroup1 peer-group
device(config-bgp-router)# neighbor PeerGroup1 description "EastCoast Neighbors"
device(config-bgp-router)# neighbor PeerGroup1 remote-as 100
device(config-bgp-router)# neighbor PeerGroup1 distribute-list out 1
device(config-bgp-router)# neighbor PeerGroup1 capability as4 enable|disable
```

The commands in this example configure a peer group called "PeerGroup1" and set the following parameters for the peer group:

- A description, "EastCoast Neighbors"
- A remote AS number, 100
- A distribute list for outbound traffic
- The capability of PeerGroup1 to utilize a four-byte AS number

The software applies these parameters to each neighbor you add to the peer group. You can override the description parameter for individual neighbors. If you set the description parameter for an individual neighbor, the description overrides the description configured for the peer group.

Syntax: `neighbor peer-group-name peer-group`

The *peer-group-name* parameter specifies the name of the group and can be up to 80 characters long. The name can contain special characters and internal blanks. If you use internal blanks, you must use quotation marks around the name. For example, the command `neighbor "My Three Peers" peer-group` is valid, but the command `neighbor My Three Peers peer-group` is not valid.

Syntax: `[no] neighbor ip-addr | peer-group-name [advertisement-interval num] [default-originate [route-map map-name]] [description string] [distribute-list { in | out } num,num... | ACL-num in | out] [ebgp-multihop [num]] [filter-list in | out num,num,... | acl-num | out | weight] [maxas-limit in [num | disable] [maximum-prefix num [threshold] [teardown]] [next-hop-self] [password string] [prefix-list string in | out] remote-as as-number] [remove-private-as] [route-map-in | out map-name] [route-reflector-client] [send-community] [soft-reconfiguration inbound] [shutdown] [timers keep-alive num hold-time num] [update-source loopback num ethernet unit/slot/port | loopback num | ve num] [weight num] [local-as as-num]`

The *ip-addr* and *peer-group-name* parameters indicate whether you are configuring a peer group or an individual neighbor. You can specify a peer group name or IP address with the **neighbor** command. If you specify a peer group name, you are configuring a peer group. If you specify a neighbor IP address, you are configuring that individual neighbor. Use the *ip-addr* parameter if you are configuring an individual neighbor instead of a peer group.

The remaining parameters are the same ones supported for individual neighbors.

Applying a peer group to a neighbor

After you configure a peer group, you can add neighbors to the group. When you add a neighbor to a peer group, you are applying all the neighbor attributes specified in the peer group to the neighbor.

To add neighbors to a peer group, enter commands such as the following.

```
device(config-bgp-router)# neighbor 192.168.1.12 peer-group PeerGroup1
device(config-bgp-router)# neighbor 192.168.2.45 peer-group PeerGroup1
device(config-bgp-router)# neighbor 192.168.3.69 peer-group PeerGroup1
```

The commands in this example add three neighbors to the peer group "PeerGroup1". As members of the peer group, the neighbors automatically receive the neighbor parameter values configured for the peer group. You also can override the parameters on an individual neighbor basis. For neighbor parameters not specified for the peer group, the neighbors use the default values.

Syntax: **[no] neighbor** *ip-addr* **peer-group** *peer-group-name*

The *ip-addr* parameter specifies the IP address of the neighbor.

The *peer-group-name* parameter specifies the peer group name.

NOTE

You must add the peer group before you can add neighbors to it.

Administratively shutting down a session with a BGP4 neighbor

You can prevent the device from starting a BGP4 session with a neighbor by administratively shutting down the neighbor. This option is very useful for situations in which you want to configure parameters for a neighbor, but are not ready to use the neighbor. You can shut the neighbor down as soon as you have added it to the device, configure the neighbor parameters, then allow the device to reestablish a session with the neighbor by removing the shutdown option from the neighbor.

When you apply the option to shut down a neighbor, the option takes place immediately and remains in effect until you remove it. If you save the configuration to the startup configuration file, the shutdown option remains in effect even after a software reload.

The software also contains an option to end the session with a BGP4 neighbor and clear the routes learned from the neighbor. Unlike this clear option, the option for shutting down the neighbor can be saved in the startup configuration file and can prevent the device from establishing a BGP4 session with the neighbor even after reloading the software.

NOTE

If you notice that a particular BGP4 neighbor never establishes a session with the device, check the running configuration and startup configuration files for that device to see whether the configuration contains a command that is shutting down the neighbor. The neighbor may have been shut down previously by an administrator.

To shut down a BGP4 neighbor, enter commands such as the following.

```
device(config)# router bgp
device(config-bgp-router)# neighbor 10.157.22.26 shutdown
device(config-bgp-router)# write memory
```

Syntax: **[no] neighbor** *ip-addr* **shutdown** [**generate-rib-out**]

The *ip-addr* parameter specifies the IP address of the neighbor.

Disabling BGP4

If you disable BGP4, the device removes all the running configuration information for the disabled protocol from the running configuration. To restore the BGP4 configuration, you must reload the software to load the BGP4 configuration from the startup configuration. When you save the startup configuration file after disabling the protocol, all of the BGP4 configuration information for the disabled protocol is removed from the startup configuration file.

The CLI displays a warning message such as the following.

```
device(config-bgp-router)# no router bgp
router bgp mode now disabled and runtime configuration is erased. All bgp config data will be lost when
writing to flash!
```

The Web Management Interface does not display a warning message.

If you are testing a BGP4 configuration and need to disable and re-enable the protocol, you should make a backup copy of the startup configuration file containing the BGP4 configuration information. If you remove the configuration information by saving the configuration after disabling the protocol, you can restore the BGP4 configuration by copying the backup copy of the startup configuration file onto the flash memory.

NOTE

To disable BGP4 without losing the BGP4 configuration information, remove the local AS (for example, by entering the **no local-as** command). When you remove the local AS, BGP4 retains the other configuration information but will not become operational until you reset the local AS.

Changing the Keep Alive Time and Hold Time

The Keep Alive Time specifies how frequently the device will send KEEPALIVE messages to its BGP4 neighbors. The Hold Time specifies how long the device will wait 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.

The default Keep Alive time is 60 seconds. The default Hold Time is 180 seconds.

NOTE

Generally, you should set the Hold Time to three times the value of the Keep Alive Time.

NOTE

You can override the global Keep Alive Time and Hold Time on individual neighbors.

To change the Keep Alive Time to 30 and Hold Time to 90, enter the following command.

```
device(config-bgp-router)# timers keep-alive 30 hold-time 90
```

Syntax: **[no] timers keep-alive** *num* **hold-time** *num*

For each keyword, *num* indicates the number of seconds. The Keep Alive Time can be 0 - 65535. The Hold Time can be 0 or 3 - 65535 (1 and 2 are not allowed). If you set the Hold Time to 0, the device waits indefinitely for messages from a neighbor without concluding that the neighbor is dead.

Changing the BGP4 next-hop update timer

By default, the device updates the BGP4 next-hop tables and affected BGP4 routes five seconds after IGP route changes. You can change the update timer to a value from 1 through 30 seconds.

To change the BGP4 update timer value to 15 seconds, for example, enter the **update-time** command at the BGP configuration level of the CLI.

```
device(config-bgp-router)# update-time 15
```

Syntax: **[no] update-time** *secs*

The `secs` parameter specifies the number of seconds and can be from 0 through 30. The default is 5. The value of 0 permits fast BGP4 convergence for situations such as link-failure or IGP route changes. Setting the value to 0 starts the BGP4 route calculation in sub-second time. All other values from 1 through 30 are still calculated in seconds.

Enabling fast external fallover

BGP4 devices rely on KEEPALIVE and UPDATE messages from neighbors to signify that the neighbors are alive. For BGP4 neighbors that are two or more hops away, such messages are the only indication that the BGP4 protocol has concerning the alive state of the neighbors. As a result, if a neighbor becomes non-operational, the device waits until the Hold Time expires or the TCP connection fails before concluding that the neighbor is not operational and closing its BGP4 session and TCP connection with the neighbor.

The device waits for the Hold Time to expire before ending the connection to a directly-attached BGP4 neighbor that becomes non-operational.

For directly-attached neighbors, the device immediately senses loss of a connection to the neighbor from a change of state of the port or interface that connects the device to the neighbor. For directly-attached EBGP neighbors, the device uses this information to immediately close the BGP4 session and TCP connection to locally attached neighbors that become non-operational.

NOTE

The fast external fallover feature applies only to directly attached EBGP neighbors. The feature does not apply to IBGP neighbors.

To enable fast external fallover, enter the following command.

```
device(config-bgp-router) # fast-external-fallover
```

To disable fast external fallover again, enter the following command.

```
device(config-bgp-router) # no fast-external-fallover
```

Syntax: `[no] fast-external-fallover`

Changing the maximum number of paths for BGP4 Multipath load sharing

Multipath load sharing enables the device to balance traffic to a route across multiple equal-cost paths of the same route type (EBGP or IBGP).

To configure the device to perform BGP4 Multipath load sharing:

- Enable IP load sharing if it is disabled.
- Set the maximum number of BGP4 load sharing paths. The default maximum number is 1, which means no BGP4 load sharing takes place by default.

NOTE

The maximum number of BGP4 load sharing paths cannot be greater than the maximum number of IP load sharing paths.

How Multipath load sharing affects route selection

During evaluation of multiple paths to select the best path to a given destination (for installment in the IP route table), the device performs a final comparison of the internal paths. The following events occur when load sharing is enabled or disabled:

- When load sharing is disabled, the device prefers the path with the lower device ID if the **compare-routerid** command is enabled.
- When load sharing and BGP4 Multipath load sharing are enabled, the device balances the traffic across multiple paths instead of choosing just one path based on device ID.

Refer to [How BGP4 selects a path for a route \(BGP best path selection algorithm\)](#) on page 299 for a description of the BGP4 algorithm.

When you enable IP load sharing, the device can load-balance BGP4 or OSPF routes across up to four equal paths by default. You can change the number load sharing paths to a value from 2 through 8.

How Multipath load sharing works

Multipath load sharing is performed in round-robin fashion and is based on the destination IP address only. The first time the device receives a packet destined for a specific IP address, the device uses a round-robin algorithm to select the path that was not used for the last newly learned destination IP address. Once the device associates a path with a particular destination IP address, the device will always use that path as long as the device contains the destination IP address in its cache.

NOTE

The device does not perform source routing. The device is concerned only with the paths to the next-hop devices, not the entire paths to the destination hosts.

A BGP4 destination can be learned from multiple BGP4 neighbors, leading to multiple BGP4 paths to reach the same destination. Each of the paths may be reachable through multiple IGP paths (multiple OSPF or RIP paths). In this case, the software installs all the multiple equal-cost paths in the BGP4 route table, up to the maximum number of BGP4 equal-cost paths allowed. The IP load sharing feature then distributes traffic across the equal-cost paths to the destination.

If an IGP path used by a BGP4 next-hop route path installed in the IP route table changes, then the BGP4 paths and IP paths are adjusted accordingly. For example, if one of the OSPF paths to reach the BGP4 next hop goes down, the software removes this path from the BGP4 route table and the IP route table. Similarly, if an additional OSPF path becomes available to reach the BGP4 next-hop device for a particular destination, the software adds the additional path to the BGP4 route table and the IP route table.

Changing the maximum number of shared BGP4 paths

To change the maximum number of BGP4 shared paths, enter commands such as the following.

```
device(config)# router bgp
device(config-bgp-router)# maximum-paths 4
device(config-bgp-router)# write memory
```

Syntax: [no] **maximum-paths** *num* | **use-load-sharing**

The *number* parameter specifies the maximum number of paths across which the device can balance traffic to a given BGP4 destination. The *number* value range is 2 through 8 and the default is 1.

When the **use-load-sharing** option is used in place of the *number* variable, the maximum IP ECMP path value is determined solely by the value configured using the **load-sharing** command.

Customizing BGP4 Multipath load sharing

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.

To enable load sharing of IBGP paths only, enter the following command at the BGP4 configuration level of the CLI.

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

To enable load sharing of EBGP paths only, enter the following command at the BGP4 configuration level of the CLI.

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

To enable load sharing of paths from different neighboring autonomous systems, enter the following command at the BGP4 configuration level of the CLI.

```
device(config-bgp)# multipath multi-as
```

Syntax: [no] multipath ebgp | ibgp | multi-as

The **ebgp**, **bgp**, and **multi-as** parameters specify the change you are making to load sharing:

- **ebgp** – Multipath load sharing applies only to EBGP paths. Multipath load sharing is disabled for IBGP paths.
- **ibgp** – Multipath load sharing applies only to IBGP paths. Multipath load sharing is disabled for EBGP paths.
- **multi-as** – Multipath load sharing is enabled for paths from different autonomous systems.

By default, load sharing applies to EBGP and IBGP paths, and does not apply to paths from different neighboring autonomous systems.

Enhancements to BGP4 Multipath load sharing

Enhancements to BGP4 Multipath load sharing allows support for load sharing of BGP4 routes in IP ECMP even if the BGP4 Multipath load sharing feature is not enabled through the **use-load-sharing** option to the **maximum-paths** command. Using the following commands, you can also set separate values for IBGP and EBGP multipath load sharing.

To set the number of equal-cost multipath IBGP routes or paths that will be selected, enter commands such as the following.

```
device(config)# router bgp
device(config-bgp)# maximum-paths ibgp
```

Syntax: [no] maximum-paths ibgp *number*

The *number* variable specifies the number of equal-cost multipath IBGP routes that will be selected. The range is 2 to 8. If the value is set to 1, BGP4 level equal-cost multipath is disabled for IBGP routes.

To set the number of equal-cost multipath EBGP routes or paths that will be selected, enter commands such as the following.

```
device(config)# router bgp
device(config-bgp)# maximum-paths ebgp
```

Syntax: [no] maximum-paths ebgp *num*

The *number* variable specifies the number of equal-cost multipath EBGP routes that will be selected. The range is 2 to 8. If the value is set to 1, BGP4 level equal-cost multipath is disabled for EBGP routes.

Specifying a list of networks to advertise

Specifying a list of networks to advertise

By default, the device sends BGP4 routes only for the networks you either identify with the **network** command or are redistributed into BGP4 from OSPF, RIP, or connected routes.

NOTE

The exact route must exist in the IP route table before the device can create a local BGP4 route.

To configure the device to advertise network 10.157.22.0/24, enter the following command.

```
device(config-bgp-router)# network 10.157.22.0 255.255.255.0
```

Syntax: **[no] network** *ip-addr ip-mask* [**route-map** *map-name*] | [**weight** *num*] | [**backdoor**]

The *ip-addr* is the network number and the *ip-mask* specifies the network mask.

The **route-map** *map-name* parameter specifies the name of the route map you want to use to set or change BGP4 attributes for the network you are advertising. The route map must already be configured. If it is not, the default action is to deny redistribution.

The **weight** *num* parameter specifies a weight to be added to routes to this network.

The **backdoor** parameter changes the administrative distance of the route to this network from the EBGp administrative distance (20 by default) to the Local BGP4 weight (200 by default), tagging the route as a backdoor route. Use this parameter when you want the device to prefer IGP routes such as RIP or OSPF routes over the EBGp route for the network.

Specifying a route map when configuring BGP4 network advertising

You can specify a route map when you configure a BGP4 network to be advertised. The device uses the route map to set or change BGP4 attributes when creating a local BGP4 route.

NOTE

You must configure the route map *before* you can specify the route map name in a BGP4 network configuration; otherwise, the route is not imported into BGP4.

To configure a route map, and use it to set or change route attributes for a network you define for BGP4 to advertise, enter commands such as the following.

```
device(config)# route-map set_net permit 1
device(config-routemap set_net)# set community no-export
device(config-routemap set_net)# exit
device(config)# router bgp
device(config-bgp)# network 10.100.1.0/24 route-map set_net
```

The first two commands in this example create a route map named "set_net" that sets the community attribute for routes that use the route map to "NO_EXPORT". The next two commands change the CLI to the BGP4 configuration level. The last command configures a network for advertising from BGP4, and associates the "set_net" route map with the network. When BGP4 originates the 10.100.1.0/24 network, BGP4 also sets the community attribute for the network to "NO_EXPORT".

Changing the default local preference

When the device uses the BGP4 algorithm to select a route to send to the IP route table, one of the parameters the algorithm uses is the local preference. Local preference indicates a degree of preference for a route relative to other routes. BGP4 neighbors can send the local preference value as an attribute of a route in an UPDATE message.

Local preference applies only to routes within the local AS. BGP4 devices can exchange local preference information with neighbors who also are in the local AS, but BGP4 devices do not exchange local preference information with neighbors in remote autonomous systems.

The default local preference is 100. For routes learned from EBGp neighbors, the default local preference is assigned to learned routes. For routes learned from IBGP neighbors, the local preference value is not changed for the route.

When the BGP4 algorithm compares routes on the basis of local preferences, the route with the higher local preference is chosen.

NOTE

To set the local preference for individual routes, use route maps.

To change the default local preference to 200, enter the following command.

```
device(config-bgp) # default-local-preference 200
```

Syntax: `[no] default-local-preference num`

The *num* parameter indicates the preference and can be a value from 0 - 4294967295.

ATTENTION

When this command is configured globally, the aggregate routes advertised through the BGP VPN do not update the local preference accordingly, even after the BGP neighbor configuration is updated by means of the **clear ip bgp neighbor all** command. You must do one of the following to resolve this:

- Execute the **clear ip bgp vrf vrf-name neighbor all** command for all associated VRFs, or
- Remove and re-add the value for **local-as** under the **router bgp** command. This stops and restarts the BGP process.

Using the IP default route as a valid next-hop for a BGP4 route

By default, the device does not use a default route to resolve a BGP4 next-hop route. If the IP 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.

In some cases, such as when the device is acting as an edge device, you can allow the device to use the default route as a valid next-hop. To do so, enter the following command at the BGP4 configuration level of the CLI.

```
device(config-bgp) # next-hop-enable-default
```

Syntax: `[no] next-hop-enable-default`

Changing the default MED (Metric) used for route redistribution

The Brocade device can redistribute directly connected routes, static IP routes, RIP routes, and OSPF routes into BGP4. The MED (metric) is a global parameter that specifies the cost that will be applied to all routes by default when they are redistributed into BGP4. When routes are selected, lower metric values are preferred over higher metric values. The default BGP4 MED value is 0 and can be assigned a value from 0 through 4294967295.

NOTE

RIP and OSPF also have default metric parameters. The parameters are set independently for each protocol and have different ranges.

To change the default metric to 40, enter the following command.

```
device(config-bgp-router) # default-metric 40
```

Syntax: `default-metric num`

The *num* indicates the metric and can be a value from 0 through 4294967295.

Enabling next-hop recursion

For each BGP4 route learned, the device performs a route lookup to obtain the IP address of the next-hop for the route. A BGP4 route is eligible for addition in the IP route table only if the following conditions are true:

- The lookup succeeds in obtaining a valid next-hop IP address for the route.
- The path to the next-hop IP address is an IGP path or a static route path.

By default, the software performs only one lookup for the next-hop IP address for the BGP4 route. If the next-hop lookup does not result in a valid next-hop IP address, or the path to the next-hop IP address is a BGP4 path, the software considers the BGP4 route destination to be unreachable. The route is not eligible to be added to the IP route table.

The BGP4 route table can contain a route with a next-hop IP 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 IGP does 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 IP 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 IP 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 IP route table. Otherwise, the device performs another lookup on the next-hop IP address of the next-hop for the next-hop gateway, and so on, until one of the lookups results in an IGP route.

NOTE

You must configure a static route or use an IGP to learn the route to the EBGP multihop peer.

Enabling recursive next-hop lookups

The recursive next-hop lookups feature is disabled by default. To enable recursive next-hop lookups, enter the following command at the BGP4 configuration level of the CLI.

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

Syntax: `[no] next-hop-recursion`

Example when recursive route lookups are disabled

The output here shows the results of an unsuccessful next-hop lookup for a BGP4 route. In this case, next-hop recursive lookups are disabled. This example is for the BGP4 route to network 10.0.0.0/24.

```
device# show ip bgp route
Total number of BGP Routes: 5
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  0.0.0.0/0      10.1.0.2      0          100          0          BI
   AS_PATH: 65001 4355 701 80
2  10.10.0.0/24    10.0.0.1      1          100          0          BI
   AS_PATH: 65001 4355 1
3  10.40.0.0/24    10.1.0.2      0          100          0          BI
   AS_PATH: 65001 4355 701 1 189
4  10.0.0.0/24    10.0.0.1      1          100          0          I
   AS_PATH: 65001 4355 3356 7170 1455
5  10.25.0.0/24    10.157.24.1   1          100          0          I
   AS_PATH: 65001 4355 701
```

In this example, the device cannot reach 10.0.0.0/24, because the next-hop IP address for the route is an IBGP route instead of an IGP route, and is considered unreachable by the device. The IP route table entry for the next-hop gateway for the BGP4 route's next-hop gateway (10.0.0.1/24) is shown here.

```
device# show ip route 10.0.0.1
Total number of IP routes: 37
Network Address  NetMask      Gateway      Port      Cost      Type
10.0.0.0         10.255.255.255  10.0.0.1    1/1/1     1          B
```

Since the route to the next-hop gateway is a BGP4 route, and not an IGP route, it cannot be used to reach 10.0.0.0/24. In this case, the device tries to use the default route, if present, to reach the subnet that contains the BGP4 route next-hop gateway.

```
device# show ip route 10.0.0.0/24
Total number of IP routes: 37
Network Address  NetMask      Gateway      Port      Cost      Type
0.0.0.0         0.0.0.0      10.0.0.202   1/1/1     1          S
```

Example when recursive route lookups are enabled

When recursive next-hop lookups are enabled, the device continues to look up the next-hop gateways along the route until the device finds an IGP route to the BGP4 route destination.

```
device# show ip bgp route
Total number of BGP Routes: 5
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  0.0.0.0/0      10.1.0.2      0          100          0          BI
   AS_PATH: 65001 4355 701 80
2  10.10.0.0/24    10.0.0.1      1          100          0          BI
   AS_PATH: 65001 4355 1
3  10.40.0.0/24    10.1.0.2      0          100          0          BI
   AS_PATH: 65001 4355 701 1 189
4  10.0.0.0/24    10.0.0.1      1          100          0          BI
   AS_PATH: 65001 4355 3356 7170 1455
5  10.25.0.0/24    10.157.24.1   1          100          0          I
   AS_PATH: 65001 4355 701
```

The first lookup results in an IBGP route, to network 10.0.0.0/24.

```
device# show ip route 10.0.0.1
Total number of IP routes: 38
```

Changing administrative distances

```

Network Address    NetMask          Gateway          Port    Cost    Type
10.0.0.0           255.255.255.0   10.0.0.1        1/1/1    1       B
AS_PATH: 65001 4355 1

```

Since the route to 10.0.0.1/24 is not an IGP route, the device cannot reach the next hop through IP, and so cannot use the BGP4 route. In this case, since recursive next-hop lookups are enabled, the device next performs a lookup for the next-hop gateway to 10.0.0.1's next-hop gateway, 10.0.0.1.

```

device# show ip bgp route 10.0.0.0
Number of BGP Routes matching display condition : 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      Metric    LocPrf    Weight    Status
1         10.0.0.0/24   1         100       0         BI
AS_PATH: 65001 4355 1

```

The next-hop IP address for 10.0.0.1 is not an IGP route, which means the BGP4 route destination still cannot be reached through IP. The recursive next-hop lookup feature performs a lookup on the next-hop gateway for 10.0.0.1

```

device# show ip route 10.0.0.1
Total number of IP routes: 38
Network Address    NetMask          Gateway          Port    Cost    Type
10.0.0.0           255.255.255.0   0.0.0.0         1/1/1    1       D
AS_PATH: 65001 4355 1 1

```

This lookup results in an IGP route that is a directly-connected route. As a result, the BGP4 route destination is now reachable through IGP, which means the BGP4 route can be added to the IP route table. The IP route table with the BGP4 route is shown here.

```

device# show ip route 10.0.0.0/24
Total number of IP routes: 38
Network Address    NetMask          Gateway          Port    Cost    Type
10.0.0.0           255.255.255.0   10.0.0.1        1/1/1    1       B
AS_PATH: 65001 4355 1

```

The device can use this route because it has an IP route to the next-hop gateway. Without recursive next-hop lookups, this route would not be in the IP route table.

Changing administrative distances

BGP4 devices can learn about networks from various protocols, including the EBGP portion of BGP4, and IGP's such as OSPF and RIP, the routes to a network may differ depending on the protocol from which the routes were learned.

To select one route over another based on the source of the route information, the device can use the administrative distances assigned to the sources. The administrative distance is a protocol-independent metric that IP devices use to compare routes from different sources.

The device re-advertises a learned best BGP4 route to neighbors even when the route table manager does not also select that route for installation in the IP route table. The best BGP4 route is the BGP4 path that BGP4 selects based on comparison of the paths' BGP4 route parameters.

When selecting a route from among different sources (BGP4, OSPF, RIP, static routes, and so on), the software compares the routes on the basis of the administrative distance for each route. If the administrative distance of the paths is lower than the administrative distance of paths from other sources (such as static IP routes, RIP, or OSPF), the BGP4 paths are installed in the IP route table.

The default administrative distances on the device are:

- Directly connected - 0 (this value is not configurable)
- Static - 1 is the default and applies to all static routes, including default routes. This can be assigned a different value.

- EBGp - 20
- OSPF - 110
- RIP - 120
- IBGP - 200
- Local BGP4 - 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 OSPF and from RIP, the device will prefer the OSPF route by default. The administrative distances are configured in different places in the software. The device re-advertises a learned best BGP4 route to neighbors by default, regardless of whether the administrative distance for the route is lower than the administrative distances of other routes from different route sources to the same destination:

- To change the EBGp, IBGP, and Local BGP4 default administrative distances, refer to the instructions in this section.
- To change the default administrative distance for OSPF, RIP, refer to [Configuring a static BGP4 network](#) on page 380.
- To change the administrative distance for static routes, refer to the instructions in this section.

To change the default administrative distances for EBGp, IBGP, and Local BGP4, enter a command such as the following.

```
device(config-bgp-router)# distance 200 200 200
```

Syntax: **[no] distance** *external-distance internal-distance local-distance*

The *external-distance* sets the EBGp distance and can be a value from 1 through 255.

The *internal-distance* sets the IBGP distance and can be a value from 1 through 255.

The *local-distance* sets the Local BGP4 distance and can be a value from 1 through 255.

Requiring the first AS to be the neighbor AS

By default, the Brocade device does not require the first AS listed in the AS_SEQUENCE field of an AS path update message from EBGp neighbors to be the AS of the neighbor that sent the update. However, you can enable the Brocade device to have this requirement. You can enable this requirement globally for the device, or for a specific neighbor or peer group. This section describes how to enable this requirement.

When you configure the device to require that the AS an EBGp neighbor is in be the same as the first AS in the AS_SEQUENCE field of an update from the neighbor, the device accepts the update only if the AS numbers match. If the AS numbers do not match, the Brocade device sends a notification message to the neighbor and closes the session. The requirement applies to all updates received from EBGp neighbors.

The hierarchy for enforcement of this feature is: a neighbor will try to use the `enforce-first-as` value if one is configured; if none is configured, the neighbor will try to use the configured value for a peer group. If neither configuration exists, enforcement is simply that of the global configuration (which is disabled by default).

To enable this feature globally, enter the **enforce-first-as** command at the BGP4 configuration level of the CLI.

```
device(config-bgp-router)# enforce-first-as
```

Syntax: **[no] enforce-first-as**

To enable this feature for a specific neighbor, enter the following command at the BGP4 configuration level.

```
device(config-bgp)# neighbor 10.1.1.1 enforce-first-as enable
```

Syntax: **[no] neighbor** *ip-address* **enforce-first-as** [**enable** | **disable**]

Disabling or re-enabling comparison of the AS-Path length

The ip-address value is the IP address of the neighbor.

When the first-as requirement is enabled, its status appears in the output of the **show running configuration** command. The optional last keyword choice of **enable** or **disable** lets you specify whether the output of the **show running configuration** command includes the configuration of the first-as requirement. This option allows the **show running configuration** command output to show what is actually configured.

To enable this feature for a peer group, enter the following command at the BGP4 configuration level.

```
device(config-bgp)# neighbor Peergroup1 enforce-first-as enable
```

Syntax: **[no] neighbor peer-group-name enforce-first-as [enable | disable]**

The *peer-group-name* value is the name of the peer group.

When the first-as requirement is enabled, its status appears in the output of the **show running configuration** command. The optional last keyword choice, that of **enable** or **disable**, lets you specify whether the output of the **show running configuration** command includes the configuration of the first-as requirement: this option helps the **show running** command output to show what you have actually configured.

The following example shows a running configuration with the first-as enforcement items (for global, peer group, and neighbor) in bold.

```
device(config)# router bgp
BGP4: Please configure 'local-as' parameter in order to enable BGP4.
device(config-bgp)# local-as 1

device(config-bgp)# enforce-first-as
device(config-bgp)# neighbor abc peer-group
device(config-bgp)# neighbor abc remote-as 2
device(config-bgp)# neighbor abc enforce-first-as disable
device(config-bgp)# neighbor 192.168.1.2 peer-group abc
device(config-bgp)# neighbor 192.168.1.2 enforce-first-as enable
```

Disabling or re-enabling comparison of the AS-Path length

AS-Path comparison is Step 5 in the algorithm that BGP4 uses to select the next path for a route. Comparison of the AS-Path length is enabled by default. To disable it, enter the following command at the BGP4 configuration level of the CLI.

```
device(config-bgp)# as-path-ignore
```

Syntax: **[no] as-path-ignore**

This command disables comparison of the AS-Path lengths of otherwise equal paths. When you disable AS-Path length comparison, the BGP4 algorithm shown in [How BGP4 selects a path for a route \(BGP best path selection algorithm\)](#) on page 299 skips from Step 4 to Step 6.

Enabling or disabling comparison of device IDs

Device ID comparison is Step 10 in the algorithm BGP4 uses to select the next path for a route.

NOTE

Comparison of device IDs is applicable only when BGP4 load sharing is disabled.

When device ID comparison is enabled, the path comparison algorithm compares the device IDs of the neighbors that sent the otherwise equal paths:

- If BGP4 load sharing is disabled (maximum-paths 1), the instructions in this section selects the path that came from the neighbor with the lower device ID.
- If BGP4 load sharing is enabled, the device load shares among the remaining paths. In this case, the device ID is not used to select a path.

NOTE

Device ID comparison is disabled by default.

To enable device ID comparison, enter the **compare-routerid** command at the BGP4 configuration level of the CLI.

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

Syntax: [no] **compare-routerid**

Configuring the device to always compare Multi-Exit Discriminators

A Multi-Exit Discriminator (MED) is a value that the BGP4 algorithm uses when it compares multiple paths received from different BGP4 neighbors in the same AS for the same route. In BGP4, a MED for a route is equivalent to its metric.

BGP4 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.

You can enable the device to always compare the MEDs, regardless of the AS information in the paths. For example, if the device receives UPDATES for the same route from neighbors in three autonomous systems, the device can compare the MEDs of all the paths together instead of comparing the MEDs for the paths in each autonomous system individually.

To enable this comparison, enter the **always-compare-med** command at the BGP4 configuration level of the CLI. This option is disabled by default.

NOTE

By default, value 0 (most favorable) is used in MED comparison when the MED attribute is not present. The default MED comparison results in the device favoring route paths that do not have their MEDs. Use the **med-missing-as-worst** command to force the device to regard a BGP4 route with a missing MED attribute as the least favorable route, when comparing the MEDs of the routes.

NOTE

MED comparison is not performed for internal routes originated within the local AS or confederation unless the **compare-med-empty-aspath** command is configured.

To configure the device to always compare MEDs, enter the following command.

```
device(config-bgp-router) # always-compare-med
```

Syntax: [no] **always-compare-med**

The following BGP4 command directs BGP4 to take the MED value into consideration even if the route has an empty as-path attribute.

```
device(config) # router bgp
device(config-bgp-router) # compare-med-empty-aspath
```

Treating missing MEDs as the worst MEDs

Syntax: [no] compare-med-empty-aspath

Treating missing MEDs as the worst MEDs

By default, the device favors a lower MED over a higher MED during MED comparison. Since the device assigns the value 0 to a route path MED if the MED value is missing, the default MED comparison results in the device favoring the route paths that are missing their MEDs.

To change this behavior so that the device favors a route that has a MED over a route that is missing its MED, enter the following command at the BGP4 configuration level of the CLI.

```
device(config-bgp-router) # med-missing-as-worst
```

Syntax: [no] med-missing-as-worst

NOTE

This command affects route selection only when route paths are selected based on MED comparison. It is still possible for a route path that is missing its MED to be selected based on other criteria. For example, a route path with no MED can be selected if its weight is larger than the weights of the other route paths.

Configuring route reflection parameters

Normally, all the BGP4 devices within an AS are fully meshed. Since each device has an IBGP session with each of the other BGP4 devices in the AS, each IBGP device has a route for each IBGP neighbor. For large autonomous systems containing many IBGP devices, the IBGP route information in each fully-meshed IBGP device may introduce too much administrative overhead.

To avoid this overhead, you can organize your IGP devices into clusters:

- A cluster is a group of IGP devices organized into route reflectors and route reflector clients. You configure the cluster by assigning a cluster ID on the route reflector and identifying the IGP neighbors that are members of that cluster. All configuration for route reflection takes place on the route reflectors. Clients are unaware that they are members of a route reflection cluster. All members of the cluster must be in the same AS. The cluster ID can be any number from 1 - 4294967295, or an IP address. The default is the device ID expressed as a 32-bit number.

NOTE

If the cluster contains more than one route reflector, you need to configure the same cluster ID on all the route reflectors in the cluster. The cluster ID helps route reflectors avoid loops within the cluster.

- A route reflector is an IGP device configured to send BGP4 route information to all the clients (other BGP4 devices) within the cluster. Route reflection is enabled on all BGP4 devices by default but does not take effect unless you add route reflector clients to the device.
- A route reflector client is an IGP device identified as a member of a cluster. You identify a device as a route reflector client on the device that is the route reflector, not on the client. The client itself requires no additional configuration. In fact, the client does not know that it is a route reflector client. The client just knows that it receives updates from its neighbors and does not know whether one or more of those neighbors are route reflectors.

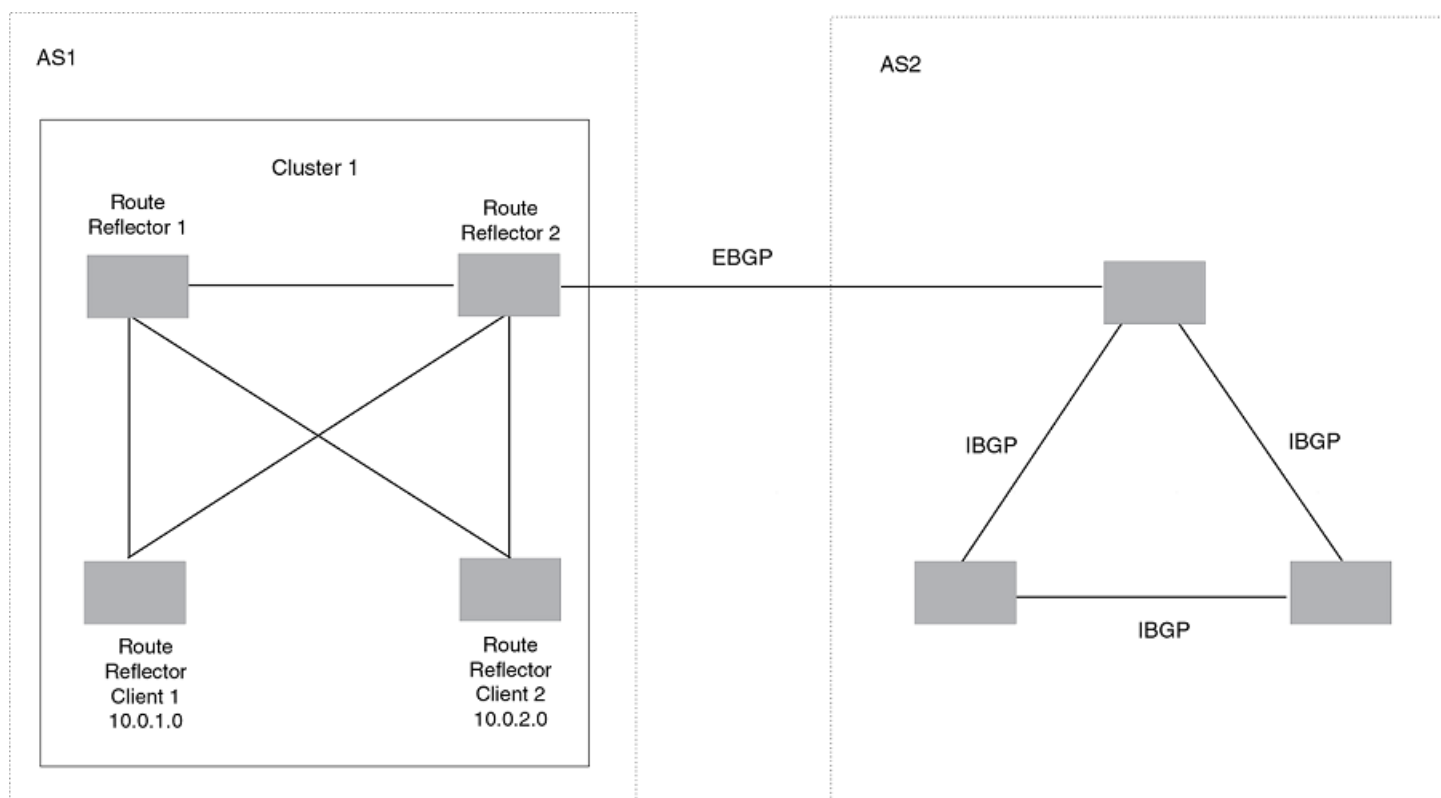
NOTE

Route reflection applies only among IBGP devices within the same AS. You cannot configure a cluster that spans multiple autonomous systems.

This is an example of a route reflector configuration. In this example, two devices are configured as route reflectors for the same cluster, which provides redundancy in case one of the reflectors becomes unavailable. Without redundancy, if a route reflector becomes unavailable, the clients for that device are cut off from BGP4 updates.

AS1 contains a cluster with two route reflectors and two clients. The route reflectors are fully meshed with other BGP4 devices, but the clients are not fully meshed and rely on the route reflectors to propagate BGP4 route updates.

FIGURE 33 A route reflector configuration



Support for RFC 4456

Route reflection on Brocade devices is based on RFC 4456. This updated RFC helps eliminate routing loops that are possible in some implementations of the older specification, RFC 1966. These instances include:

- The device adds the route reflection attributes only if it is a route reflector, and only when advertising IBGP route information to other IBGP neighbors. The attributes are not used when communicating with EBGP neighbors.
- A device configured as a route reflector sets the ORIGINATOR_ID attribute to the device ID of the device that originated the route. The route reflector sets this attribute only if this is the first time the route is being reflected (sent by a route reflector).
- If a device receives a route with an ORIGINATOR_ID attribute value that is the same as the ID of the device, the device discards the route and does not advertise it. By discarding the route, the device prevents a routing loop.
- The first time a route is reflected by a device configured as a route reflector, the route reflector adds the CLUSTER_LIST attribute to the route. Other route reflectors that receive the route from an IBGP neighbor add their cluster IDs to the front of the routes CLUSTER_LIST. If the route reflector does not have a cluster ID configured, the device adds its device ID to the front of the CLUSTER_LIST.

- If a device configured as a route reflector receives a route with a CLUSTER_LIST that contains the cluster ID of the route reflector, the route reflector discards the route.

Configuration procedures for BGP4 route reflector

To configure a Brocade device to be a BGP4 route reflector, use either of the following methods.

NOTE

All configuration for route reflection takes place on the route reflectors, not on the clients.

Enter the following commands to configure a Brocade device as route reflector 1. To configure route reflector 2, enter the same commands on the device that will be route reflector 2. The clients require no configuration for route reflection.

```
device(config-bgp) # cluster-id 1
```

Syntax: [no] **cluster-id** *num* | *ip-addr*

The *num* and *ip-addr* parameters specify the cluster ID and can be a number from 1 - 4294967295, or an IP address. The default is the device ID. You can configure one cluster ID on the device. All route-reflector clients for the device are members of the cluster.

NOTE

If the cluster contains more than one route reflector, you need to configure the same cluster ID on all the route reflectors in the cluster. The cluster ID helps route reflectors avoid loops within the cluster.

To add an IBGP neighbor to the cluster, enter the following command:

```
device(config-bgp) # neighbor 10.0.1.0 route-reflector-client
```

Syntax: [no] **neighbor** *ip-addr* **route-reflector-client**

Disabling or re-enabling client-to-client route reflection

By default, the clients of a route reflector are not required to be fully meshed. Routes from a client are reflected to other clients. However, if the clients are fully meshed, route reflection is not required between clients.

If you need to disable route reflection between clients, enter the **no client-to-client-reflection** command. When this feature is disabled, route reflection does not occur between clients does still occur between clients and non-clients.

```
device(config-bgp-router) # no client-to-client-reflection
```

Enter the following command to re-enable the feature.

```
device(config-bgp) # client-to-client-reflection
```

Syntax: [no] **client-to-client-reflection**

Configuring confederations

A **confederation** is a BGP4 Autonomous System (AS) that has been subdivided into multiple, smaller autonomous systems. Subdividing an AS into smaller autonomous systems simplifies administration and reduces BGP4-related traffic, which in turn reduces the complexity of the Interior Border Gateway Protocol (IBGP) mesh among the BGP4 devices in the AS.

The Brocade implementation of this feature is based on RFC 3065.

Normally, all BGP4 devices within an AS must be fully meshed, so that each BGP4 device has BGP4 sessions to all the other BGP4 devices within the AS. This is feasible in smaller autonomous systems, but becomes unmanageable in autonomous systems containing many BGP4 devices.

When you configure BGP4 devices into a confederation, all the devices within a sub-AS (a subdivision of the AS) use IBGP and must be fully meshed. However, devices use EBGP to communicate between different sub-autonomous systems.

NOTE

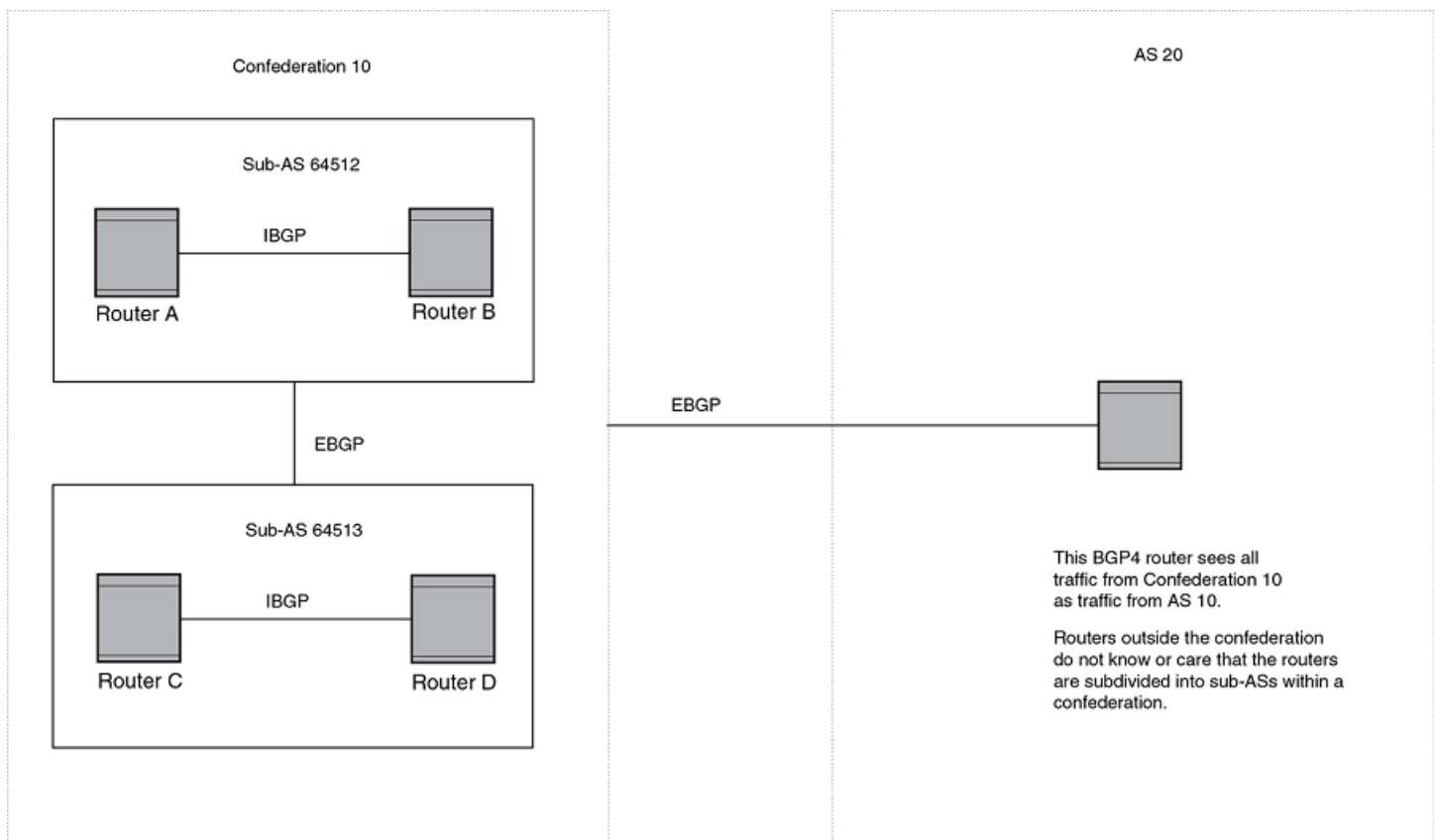
Another way to reduce the complexity of an IBGP mesh is to use route reflection. However, if you want to run different Interior Gateway Protocols (IGPs) within an AS, you must configure a confederation. You can run a separate IGP within each sub-AS.

To configure a confederation, configure groups of BGP4 devices into sub-autonomous systems. A sub-AS is simply an AS. The term "sub-AS" distinguishes autonomous systems within a confederation from autonomous systems that are not in a confederation. For the viewpoint of remote autonomous systems, the confederation ID is the AS ID. Remote autonomous systems do not know that the AS represents multiple sub-autonomous systems with unique AS IDs.

NOTE

You can use any valid AS numbers for the sub-autonomous systems. If your AS is connected to the Internet, Brocade recommends that you use numbers from within the private AS range (64512 through 65535). These are private autonomous system numbers and BGP4 devices do not propagate these AS numbers to the Internet.

FIGURE 34 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 a BGP4 confederation

To configure a BGP4 configuration, perform these configuration tasks on each BGP4 device within the confederation:

- Configure the local AS number. The local AS number indicates membership in a sub-AS. All BGP4 devices with the same local AS number are members of the same sub-AS. BGP4 devices use the local AS number when communicating with other BGP4 devices in the confederation.
- Configure the confederation ID. The confederation ID is the AS number by which BGP4 devices outside the confederation recognize the confederation. A BGP4 device outside the confederation is not aware of, and does not care that BGP4 devices are in multiple sub-autonomous systems. A BGP4 device uses the confederation ID to communicate with devices outside the confederation. The confederation ID must differ from the sub-AS numbers.
- Configure the list of the sub-AS numbers that are members of the confederation. All devices within the same sub-AS use IBGP to exchange device information. Devices in different sub-autonomous systems within the confederation use EBGP to exchange device information.

To configure four devices to be members of confederation 10 (consisting of sub-autonomous systems 64512 and 64513), enter commands such as the following.

Commands for device A

```
deviceA(config)# router bgp
deviceA(config-bgp-router)# local-as 64512
deviceA(config-bgp-router)# confederation identifier 10
deviceA(config-bgp-router)# confederation peers 64512 64513
deviceA(config-bgp-router)# write memory
```

Syntax: [no] local-as *num*

The *num* parameter with the **local-as** command indicates the AS number for the BGP4 devices within the sub-AS. You can specify a number in the range 1 - 4294967295. Brocade recommends that you use a number within the range of well-known private autonomous systems, 64512 through 65535.

Syntax: [no] confederation identifier *num*

The *num* parameter with the **confederation identifier** command indicates the confederation number. The confederation ID is the AS number by which BGP4 devices outside the confederation recognize the confederation. A BGP4 device outside the confederation is not aware of, and does not care that your BGP4 devices are in multiple sub-autonomous systems. BGP4 devices use the confederation ID when communicating with devices outside the confederation. The confederation ID must be different from the sub-AS numbers. For the *num* parameter, you can specify a number in the range 1 - 4294967295.

Syntax: [no] confederation peers *num* [*num* ...]

The *num* parameter with the **confederation peers** command indicates the sub-AS numbers for the sub-autonomous systems in the confederation. You can list all sub-autonomous systems in the confederation. You must specify all the sub-autonomous systems with which this device has peer sessions in the confederation. All the devices within the same sub-AS use IBGP to exchange device

information. Devices in different sub-autonomous systems within the confederation use EBGp to exchange device information. The *num* is a number in the range 1 - 4294967295.

Commands for device B

```
deviceB(config)# router bgp
deviceB(config-bgp-router)# local-as 64512
deviceB(config-bgp-router)# confederation identifier 10
deviceB(config-bgp-router)# confederation peers 64512 64513
deviceB(config-bgp-router)# write memory
```

Commands for device C

```
deviceC(config)# router bgp
deviceC(config-bgp-router)# local-as 64513
deviceC(config-bgp-router)# confederation identifier 10
deviceC(config-bgp-router)# confederation peers 64512 64513
deviceC(config-bgp-router)# write memory
```

Commands for device D

```
deviceD(config)# router bgp
deviceD(config-bgp-router)# local-as 64513
deviceD(config-bgp-router)# confederation identifier 10
deviceD(config-bgp-router)# confederation peers 64512 64513
deviceD(config-bgp-router)# write memory
```

Aggregating routes advertised to BGP4 neighbors

By default, the device advertises individual routes for all networks. The aggregation feature allows you to configure the device to aggregate routes from a range of networks into a single network prefix. For example, without aggregation, the device will individually advertise routes for networks 10.95.1.0/24, 10.95.2.0/24, and 10.95.3.0/24. You can configure the device to end a single, aggregate route for the networks instead. The aggregate route can be advertised as 10.95.0.0/16.

To aggregate routes for 10.157.22.0/24, 10.157.23.0/24, and 10.157.24.0/24, enter the following command.

```
device(config-bgp)# aggregate-address 10.157.0.0 255.255.0.0
```

Syntax: [no] **aggregate-address** *ip-addr ip-mask* [**as-set**] [**summary-only**] [**suppress-map** *map-name*] [**advertise-map** *map-name*] [**attribute-map** *map-name*]

The *ip-addr* and *ip-mask* parameters specify the aggregate value for the networks. Specify 0 for the host portion and for the network portion that differs among the networks in the aggregate. For example, to aggregate 10.0.1.0/24, 10.0.2.0/24, and 10.0.3.0/24, enter the IP address 10.0.0.0 and the network mask 255.255.0.0.

The **as-set** parameter causes the device to aggregate AS-path information for all the routes in the aggregate address into a single AS-path.

The **summary-only** parameter prevents the device from advertising more specific routes contained within the aggregate route.

The **suppress-map** *map-name* parameter prevents the more specific routes contained in the specified route map from being advertised.

The **advertise-map** *map-name* parameter configures the device to advertise the more specific routes in the specified route map.

The **attribute-map** *map-name* parameter configures the device to set attributes for the aggregate routes based on the specified route map.

NOTE

For the **suppress-map** , **advertise-map** , and **attribute-map** parameters, the route map must already be defined.

Configuring BGP4 restart

BGP4 restart can be configured for a global routing instance or for a specified Virtual Routing and Forwarding (VRF) instance. The following sections describe how to enable the BGP4 restart feature.

BGP4 restart is enabled by default.

Configuring BGP4 Restart for the global routing instance

Use the following command to enable the BGP4 Restart feature globally on a device.

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

Syntax: **[no]** graceful-restart

Configuring BGP4 Restart for a VRF

Use the following command to enable the BGP4 Restart feature for a specified VRF.

```
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast vrf blue
device(config-bgp-ipv4u-vrf)# graceful-restart
```

Syntax: **[no]** graceful-restart

Configuring timers for BGP4 Restart (optional)

You can optionally configure the following timers to change their values from the default values:

- Restart Timer
- Stale Routes Timer
- Purge Timer

The *seconds* variable sets the maximum restart wait time advertised to neighbors. Possible values are 1- 3600 seconds. The default value is 120 seconds.

Configuring the restart timer for BGP4 Restart

Use the following command to specify the maximum amount of time a device will maintain routes from and forward traffic to a restarting device.

```
device(config-bgp)# graceful-restart restart-time 150
```

Syntax: **[no]** graceful-restart restart-time *seconds*

The *seconds* variable sets the maximum restart wait time advertised to neighbors. Possible values are 1 through 3600 seconds. The default value is 120 seconds.

Configuring BGP4 Restart stale routes timer

Use the following command to specify the maximum amount of time a helper device will wait for an end-of-RIB message from a peer before deleting routes from that peer.

```
device(config-bgp)# graceful-restart stale-routes-time 120
```

Syntax: [no] graceful-restart stale-routes-time *seconds*

The *seconds* variable sets the maximum time before a helper device cleans up stale routes. Possible values are 1 through 3600 seconds. The default value is 360 seconds.

Configuring BGP4 Restart purge timer

Use the following command to specify the maximum amount of time a device will maintain stale routes in its routing table before purging them.

```
device(config-bgp)# graceful-restart purge-time 900
```

Syntax: [no] graceful-restart purge-time *seconds*

The *seconds* variable sets the maximum time before a restarting device cleans up stale routes. Possible values are 1 - 3600 seconds. The default value is 600 seconds.

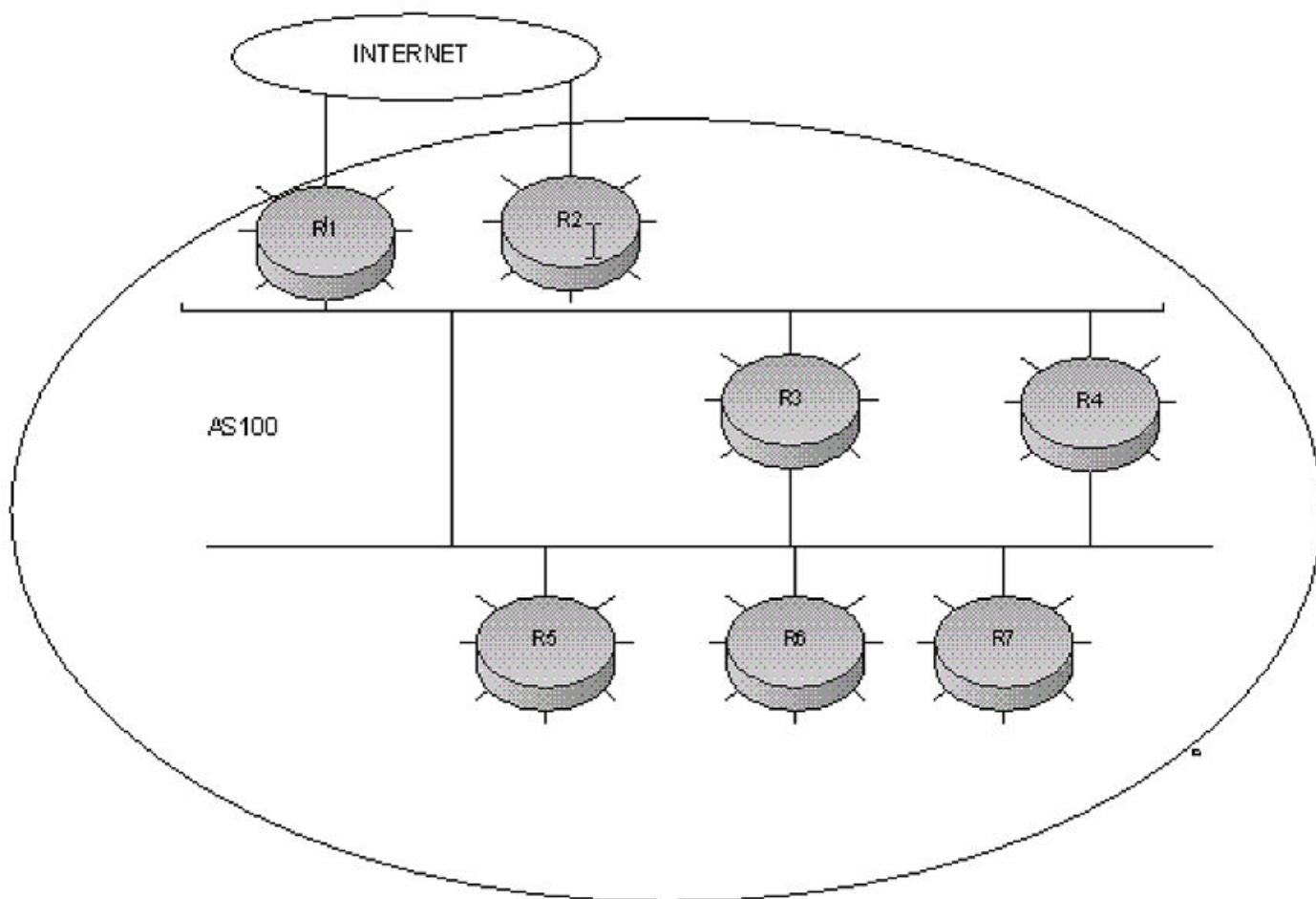
BGP4 null0 routing

BGP4 considers the null0 route in the routing table (for example, static route) as a valid route, and can use the null0 route to resolve the next hop. If the next hop for BGP4 resolves into a null0 route, the BGP4 route is also installed as a null0 route in the routing table.

The null0 routing feature allows network administrators to block certain network prefixes using null0 routes and route-maps, directing a remote device to drop all traffic for a network prefix by redistributing a null0 route into BGP4.

This example shows a topology for a null0 routing application example.

FIGURE 35 SAMPLE null0 routing application



Configuring BGP4 null0 routing

The following example configures a null0 routing application to stop denial of service attacks from remote hosts on the Internet.

1. Select a device, for example, device 6, to distribute null0 routes throughout the BGP4 network.
2. To configure a route-map perform the following step.
 - Configure a route-map to match a particular tag (50) and set the next-hop address to an unused network address (10.199.1.1).
3. Set the local-preference to a value higher than any possible internal or external local-preference (50).
4. Complete the route map by setting origin to IGP.
5. On device 6, redistribute the static routes into BGP4, using route-map *route-map-name* (redistribute static route-map block user).
6. To configure a route-map perform the following step.
 - On device 1, (the device facing the Internet), configure a null0 route matching the next-hop address in the route-map (ip route 10.199.1.1/32 null0).

7. Repeat step 3 for all devices interfacing with the Internet (edge corporate devices). In this case, device 2 has the same null0 route as device 1.
8. On device 6, configure the network prefixes associated with the traffic you want to drop. The static route IP address references a destination address. You must point the static route to the egress port, (for example, Ethernet 1/3/7), and specify the tag 50, matching the route-map configuration.

Configuration examples

Device 6

The following configuration defines specific prefixes to filter:

```
device(config)# ip route 10.0.0.40/29 ethernet 1/3/7 tag 50
device(config)# ip route 10.0.0.192/27 ethernet 1/3/7 tag 50
device(config)# ip route 10.014.0/23 ethernet 1/3/7 tag 50
```

The following configuration redistributes routes into BGP4.

```
device(config)# router bgp
device(config-bgp-router)# local-as 100
device(config-bgp-router)# neighbor router1_int_ip address remote-as 100
device(config-bgp-router)# neighbor router2_int_ip address remote-as 100
device(config-bgp-router)# neighbor router3_int_ip address remote-as 100
device(config-bgp-router)# neighbor router4_int_ip address remote-as 100
device(config-bgp-router)# neighbor router5_int_ip address remote-as 100
device(config-bgp-router)# neighbor router7_int_ip address remote-as 100
device(config-bgp-router)# redistribute static route-map blockuser
device(config-bgp-router)# exit
```

The following configuration defines the specific next hop address and sets the local preference to preferred.

```
device(config)# route-map blockuser permit 10
device(config-routemap blockuser)# match tag 50
device(config-routemap blockuser)# set ip next-hop 10.199.1.1
device(config-routemap blockuser)# set local-preference 1000000
device(config-routemap blockuser)# set origin igp
device(config-routemap blockuser)# exit
```

NOTE

A match tag can take up to 16 tags. During the execution of a route-map, a match on any tag value in the list is considered a successful match.

Device 1

The following configuration defines the null0 route to the specific next hop address. The next hop address 10.199.1.1 points to the null0 route.

```
device(config)# ip route 10.199.1.1/32 null0
device(config)# router bgp
device(config-bgp-router)# local-as 100
device(config-bgp-router)# neighbor router2_int_ip address remote-as 100
device(config-bgp-router)# neighbor router3_int_ip address remote-as 100
device(config-bgp-router)# neighbor router4_int_ip address remote-as 100
device(config-bgp-router)# neighbor router5_int_ip address remote-as 100
device(config-bgp-router)# neighbor router6_int_ip address remote-as 100
device(config-bgp-router)# neighbor router7_int_ip address remote-as 100
```

Device 2

Configuring BGP4 restart

The following configuration defines a null0 route to the specific next hop address. The next hop address 10.199.1.1 points to the null0 route, which gets blocked.

```
device(config)# ip route 10.199.1.1/32 null0
device(config)# router bgp
device(config-bgp-router)# local-as 100
device(config-bgp-router)# neighbor router1_int_ip address remote-as 100
device(config-bgp-router)# neighbor router3_int_ip address remote-as 100
device(config-bgp-router)# neighbor router4_int_ip address remote-as 100
device(config-bgp-router)# neighbor router5_int_ip address remote-as 100
device(config-bgp-router)# neighbor router6_int_ip address remote-as 100
device(config-bgp-router)# neighbor router7_int_ip address remote-as 100
```

Show commands for BGP4 null 0 routing

After configuring the null0 application, you can display the output using **show** commands.

Device 6

Show ip route static output for device 6.

```
device# show ip route static
Type Codes - B:BGp D:Connected S:Static R:RIP O:OSPF; Cost - Dist/Metric
      Destination      Gateway      Port      Cost      Type
1      10.0.0.40/29      DIRECT      eth 1/3/7  1/1      S
2      10.0.0.192/27     DIRECT      eth 1/3/7  1/1      S
3      10.0.14.0/23      DIRECT      eth 1/3/7  1/1      S
device#
```

Device 1 and 2

Show ip route static output for device 1 and device 2.

```
device# show ip route static
Type Codes - B:BGp D:Connected S:Static R:RIP O:OSPF; Cost - Dist/Metric
      Destination      Gateway      Port      Cost      Type
1      10.199.1.1/32     DIRECT      drop      1/1      S
device#
```

Device 6

The following is the **show ip bgp route** output for Device-6

```
device# show ip bgp route
Total number of BGP Routes: 126
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
      E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
      S:SUPPRESSED F:FILTERED s:STALE
      Prefix      Next Hop      MED      LocPrf      Weight      Status
1      10.0.1.0/24      10.0.1.3      0      100      0      BI
      AS_PATH:
      .
9      10.0.0.16/30      10.0.1.3      .      100      0      I
      AS_PATH: 85
10     10.0.0.40/29      10.199.1.1/32 1      1000000 32768      BL
      AS_PATH:
11     10.0.0.80/28      10.0.1.3      .      100      0      I
      .
36     10.0.0.96/28      10.0.1.3      .      100      0      I
      AS_PATH: 50
37     10.0.0.192/27     10.199.1.1/32 1      1000000 32768      BL
      AS_PATH:
64     10.0.7.0/24      10.0.1.3      .      100      0      I
      AS_PATH: 10
65     10.0.14.0/23      10.199.1.1/32 1      1000000 32768      BL
      AS_PATH: ..
```


Device 1 and 2

The **show ip route** output for device 1 and device 2 shows "drop" under the Port column for the network prefixes you configured with null0 routing

```
device# show ip route
Total number of IP routes: 133
Type Codes - B:BGP D:Connected S:Static R:RIP O:OSPF; Cost - Dist/Metric
  Destination Gateway Port Cost Type
1 10.0.1.24/32 DIRECT loopback 1 0/0 D
2 10.0.1.0/24 DIRECT eth 1/2/7 0/0 D
3 10.0.1.1/24 DIRECT eth 1/2/1 0/0 D
.
13 10.0.0.6/31 10.0.1.3 eth 1/2/2 20/1 B
14 10.0.0.16/30 10.0.1.3 eth 1/2/2 20/1 B
15 10.0.0.40/29 DIRECT drop 200/0 B
.
42 10.0.0.192/27 DIRECT drop 200/0 B
43 10.0.1.128/26 10.0.1.3 eth 1/2/7 20/1 B
.
69 10.0.7.0/24 10.0.1.3 eth 1/2/10 20/1 B
70 10.0.14.0/23 DIRECT drop 200/0 B
.
131 10.144.0.0/12 10.0.1.3 eth 1/3/4 20/1 B
132 10.199.1.1/32 DIRECT drop 1/1
```

Modifying redistribution parameters

By default, the route information between BGP4 and the IP IGP (RIP and OSPF) is not redistributed. You can configure the device to redistribute OSPF routes, RIP routes, directly connected routes, or static routes into BGP4.

To enable redistribution of all OSPF routes and directly attached routes into BGP4, enter the following commands.

```
device(config)# router bgp
device(config-bgp-router)# redistribute ospf
device(config-bgp-router)# redistribute connected
device(config-bgp-router)# write memory
```

Syntax: **[no] redistribute connected | ospf | rip | static**

The **connected** parameter indicates that you are redistributing routes to directly attached devices into BGP4.

The **ospf** parameter indicates that you are redistributing OSPF routes into BGP4.

NOTE

Entering **redistribute ospf** simply redistributes internal OSPF routes. To redistribute external OSPF routes also, use the **redistribute ospf match external** command.

The **rip** parameter indicates that you are redistributing RIP routes into BGP4.

The **isis** parameter indicates that you are redistributing IS-IS routes into BGP4.

The **static** parameter indicates that you are redistributing static routes into BGP4.

Redistributing connected routes

To configure BGP4 to redistribute directly connected routes, enter the following command.

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

Syntax: **[no] redistribute connected [metric num] [route-map map-name]**

The **connected** parameter indicates that you are redistributing routes to directly attached devices into BGP4.

The **metric num** parameter changes the metric. You can specify a value from 0 through 4294967295. The default is not assigned.

The **route-map map-name** parameter specifies a route map to be consulted before adding the RIP route to the BGP4 route table.

NOTE

The route map you specify must already be configured on the device.

Redistributing RIP routes

To configure BGP4 to redistribute RIP routes and add a metric of 10 to the redistributed routes, enter the following command.

```
device(config-bgp-router)# redistribute rip metric 10
```

Syntax: [no] redistribute rip [metric num] [route-map map-name]

The **rip** parameter indicates that you are redistributing RIP routes into BGP4.

The **metric num** parameter changes the metric. You can specify a value from 0 - 4294967295. The default is not assigned.

The **route-map map-name** parameter specifies a route map to be consulted before adding the RIP route to the BGP4 route table.

NOTE

The route map you specify must already be configured on the device.

Redistributing OSPF external routes

To configure the device to redistribute OSPF external type 1 routes, enter the following command.

```
device(config-bgp-router)# redistribute ospf match external1
```

Syntax: [no] redistribute ospf [match internal | external1 | external2] [metric num] [route-map map-name]

The **ospf** parameter indicates that you are redistributing OSPF routes into BGP4.

The match **internal**, **external1**, and **external2** parameters apply only to OSPF. These parameters specify the types of OSPF routes to be redistributed into BGP4. The default is internal.

NOTE

If you do not enter a value for the **match** parameter, (for example, you enter **redistribute ospf** only) then only internal OSPF routes will be redistributed.

The **metric num** parameter changes the metric. You can specify a value from 0 through 4294967295. The default is not assigned.

The **route-map map-name** parameter specifies a route map to be consulted before adding the OSPF route to the BGP4 route table.

NOTE

The route map you specify must already be configured on the device.

NOTE

If you use both the **redistribute ospf route-map** command and the **redistribute ospf match internal** command, the software uses only the route map for filtering.

Redistributing static routes

To configure the device to redistribute static routes, enter the following command.

```
device(config-bgp) # redistribute static
```

Syntax: `[no] redistribute static [metric num] [route-map map-name]`

The **static** parameter indicates that you are redistributing static routes into BGP4.

The **metric***num* parameter changes the metric. You can specify a value from 0 - 4294967295. The default is 0.

The **route-map** *map-name* parameter specifies a route map to be consulted before adding the static route to the BGP4 route table.

NOTE

The route map you specify must already be configured on the device.

Redistributing IBGP routes

By default, the device does not allow redistribute IBGP routes from BGP4 into RIP, or OSPF. This behavior helps eliminate routing loops. In non-default VRF instances, by default, the device does allow redistribution IBGP routes from BGP4 into RIP, OSPF.

To enable the device to redistribute BGP4 routes into OSPF and RIP, enter the following command.

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

Syntax: `[no] bgp-redistribute-internal`

To disable redistribution of IBGP routes into RIP, IS-IS, and OSPF, enter the **bgp-redistribute-internal** command.

Filtering

This section describes the following:

- AS-path filtering
- Defining and applying IP prefix lists
- Defining neighbor distribute lists
- Defining route maps
- Configuring cooperative BGP4 route filtering

AS-path filtering

You can filter updates received from BGP4 neighbors based on the contents of the AS-path list accompanying the updates. For example, to deny routes that have the AS 10.3.2.1 in the AS-path from entering the BGP4 route table, you can define a filter.

The device provides the following methods for filtering on AS-path information:

- AS-path filters
- AS-path ACLs

NOTE

The device cannot support AS-path filters and AS-path ACLs at the same time. Use one method or the other, but do not mix methods.

NOTE

Once you define a filter or ACL, the default action for updates that do not match a filter is **deny**. To change the default action to **permit**, configure the last filter or ACL as **permit any any**.

AS-path filters or AS-path ACLs can be referred to by the filter list number of a BGP4 neighbor as well as by match clauses in a route map.

Defining an AS-path ACL

To configure an AS-path list that uses "acl 1", enter a command such as the following.

```
device(config)# ip as-path access-list acl1 permit 100
device(config)# router bgp
device(config-bgp-router)# neighbor 10.10.10.1 filter-list acl1 in
```

Syntax: [no] ip as-path access-list *string* [seq *s* *eq-value*] deny | permit *regular-expression*

The **ip as-path** command configures an AS-path ACL that permits routes containing AS number 100 in their AS paths. The **neighbor** command then applies the AS-path ACL to advertisements and updates received from neighbor 10.10.10.1. In this example, the only routes the device permits from neighbor 10.10.10.1 are those whose AS-paths contain AS-path number 100.

The *string* parameter specifies the ACL name. (If you enter a number, the CLI interprets the number as a text string.)

The **seqseq-value** parameter is optional and specifies the sequence number for the AS-path list. If you do not specify a sequence number, the software numbers in increments of 5, beginning with number 5. The software interprets the entries in an AS-path list in numerical order, beginning with the lowest sequence number.

The **deny** and **permit** parameters specify the action the software takes if the AS-path list for a route matches a match clause in this ACL. To configure the AS-path match clauses in a route map, use the match as-path command.

The *regular-expression* parameter specifies the AS path information you want to permit or deny to routes that match any of the match clauses within the ACL. You can enter a specific AS number or use a regular expression.

The **neighbor** command uses the **filter-list** parameter to apply the AS-path ACL to the neighbor.

Using regular expressions

Use a regular expression for the *as-path* parameter to specify a single character or multiple characters as a filter pattern. If the AS-path matches the pattern specified in the regular expression, the filter evaluation is true; otherwise, the evaluation is false.

You can also include special characters that influence the way the software matches the AS-path against the filter value.

To filter on a specific single-character value, enter the character for the *as-path* parameter. For example, to filter on AS-paths that contain the letter "z", enter the following command:

```
device(config-bgp-router)# ip as-path access-list acl1 permit z
```

To filter on a string of multiple characters, enter the characters in brackets. For example, to filter on AS-paths that contain "x", "y", or "z", enter the following command.

```
device(config-bgp-router)# ip as-path access-list acl1 permit [xyz]
```

BGP4 Special characters

When you enter a single-character expression or a list of characters, you also can use the special characters listed in "Using regular expressions." The description for each character includes an example. Some special characters must be placed in front of the characters they control and others must be placed after the characters they control. The examples show where to place the special character.

TABLE 45 BGP4 special characters for regular expressions

Character	Operation
.	The period matches on any single character, including a blank space. For example, the following regular expression matches for "aa", "ab", "ac", and so on, but not just "a". a.
*	The asterisk matches on zero or more sequences of a pattern. For example, the following regular expression matches on an AS-path that contains the string "1111" followed by any value: 1111*
+	The plus sign matches on one or more sequences of a pattern. For example, the following regular expression matches on an AS-path that contains a sequence of "g"s, such as "deg", "degg", "deggg", and so on: deg+
?	The question mark matches on zero occurrences or one occurrence of a pattern. For example, the following regular expression matches on an AS-path that contains "dg" or "deg": de?g
^	A caret (when not used within brackets) matches on the beginning of an input string. For example, the following regular expression matches on an AS-path that begins with "3": ^3
\$	A dollar sign matches on the end of an input string. For example, the following regular expression matches on an AS-path that ends with "deg": deg\$
_	An underscore matches on one or more of the following: <ul style="list-style-type: none"> , (comma) { (left curly brace) } (right curly brace) ((left parenthesis)) (right parenthesis) The beginning of the input string The end of the input string A blank space For example, the following regular expression matches on "100" but not on "1002", "2100", and so on. _100_
[]	Square brackets enclose a range of single-character patterns. For example, the following regular expression matches on an AS-path that contains "1", "2", "3", "4", or "5": [1-5] You can use the following expression symbols within the brackets. These symbols are allowed only inside the brackets: <ul style="list-style-type: none"> ^ - The caret matches on any characters except the ones in the brackets. For example, the following regular expression matches on an AS-path that does not contain "1", "2", "3", "4", or "5": [^1-5]

TABLE 45 BGP4 special characters for regular expressions (continued)

Character	Operation
	<ul style="list-style-type: none"> - The hyphen separates the beginning and ending of a range of characters. A match occurs if any of the characters within the range is present. Refer to the example above.
	<p>A vertical bar (sometimes called a pipe or a "logical or") separates two alternative values or sets of values. The AS-path can match one or the other value. For example, the following regular expression matches on an AS-path that contains either "abc" or "defg":</p> <p>(abc) (defg)</p> <p>NOTE The parentheses group multiple characters to be treated as one value. Refer to the following row for more information about parentheses.</p>
()	<p>Parentheses allow you to create complex expressions. For example, the following complex expression matches on "abc", "abcabc", or "abcabcabcdefg", but not on "abcdefgdefg":</p> <p>((abc)+)((defg)?)</p>

To filter for a special character instead of using the special character as described in "Using regular expressions," enter "\" (backslash) in front of the character. For example, to filter on AS-path strings containing an asterisk, enter the asterisk portion of the regular expression as "*".

```
device(config-bgp-router)# ip as-path access-list acl2 deny \*
```

To use the backslash as a string character, enter two slashes. For example, to filter on AS-path strings containing a backslash, enter the backslash portion of the regular expression as "\\".

```
device(config-bgp-router)# ip as-path access-list acl2 deny \\\
```

BGP4 filtering communities

You can filter routes received from BGP4 neighbors based on community names.

A community is an optional attribute that identifies the route as a member of a user-defined class of routes. Community names are arbitrary values made of two five-digit integers joined by a colon. You determine what the name means when you create the community name as a route attribute. Each string in the community name can be a number from 0 through 65535.

This format allows you to easily classify community names. For example, a common convention used in community naming is to configure the first string as the local AS and the second string as the unique community within that AS. Using this convention, communities 1:10, 1:20, and 1:30 can be easily identified as member communities of AS 1.

The device provides the following methods for filtering on community information.

- Community filters
- Community list ACLs

NOTE

The device cannot actively support community filters and community list ACLs at the same time. Use one method or the other but do not mix methods.

NOTE

Once you define a filter or ACL, the default action for communities that do not match a filter or ACL is **deny**. To change the default action to **permit**, configure the last filter or ACL entry as **permit any any**.

Community filters or ACLs can be referred to by match clauses in a route map.

Defining a community ACL

To configure community ACL 1, enter a command such as the following. This command configures a community ACL that permits routes that contain community 123:2.

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

Syntax: `no ip community-list standard string [seq seq-value] deny | permit community-num`

The *string* parameter specifies the ACL name. (If you enter a number, the CLI interprets the number as a text string.)

The **standard** parameter specifies whether you are configuring a standard community ACL.

The **seq seq-value** parameter is optional and specifies the sequence number for the community list. You can configure up to 199 entries in a community list. If you do not specify a sequence number, the software numbers the entries in increments of 5, beginning with number 5. The software interprets the entries in a community list in numerical order, beginning with the lowest sequence number.

The **deny** and **permit** parameters specify the action the software takes if a route community list matches a match clause in this ACL. To configure the community-list match clauses in a route map, use the **match community** command.

The *community-num* parameter specifies the community type or community number. This parameter can have the following values:

- **num:num** - A specific community number
- **internet** - The Internet community
- **no-export** - The community of sub-autonomous systems within a confederation. Routes with this community can be exported to other sub-autonomous systems within the same confederation but cannot be exported outside the confederation to other autonomous systems or otherwise sent to EBGp neighbors.
- **local-as** - The local sub-AS within the confederation. Routes with this community can be advertised only within the local subAS.
- **no-advertise** - Routes with this community cannot be advertised to any other BGP4 devices at all.

The *regular-expression* parameter specifies a regular expression for matching on community names.

To use a community-list filter, use route maps with the **match community** parameter.

Defining and applying IP prefix lists

An IP prefix list specifies a list of networks. When you apply an IP prefix list to a neighbor, the device sends or receives only a route whose destination is in the IP prefix list. The software interprets the prefix lists in order, beginning with the lowest sequence number.

To configure an IP prefix list and apply it to a neighbor, enter commands such as the following.

```
device(config)# ip prefix-list Routesfor20 permit 10.20.0.0/24
device(config)# router bgp
device(config-bgp-router)# neighbor 10.10.10.1 prefix-list Routesfor20 out
```

These commands configure an IP prefix list named Routesfor20, which permits routes to network 10.20.0.0/24. The **neighbor** command configures the device to use IP prefix list Routesfor20 to determine which routes to send to neighbor 10.10.10.1. The device sends routes that go to 10.20.x.x to neighbor 10.10.10.1 because the IP prefix list explicitly permits these routes to be sent to the neighbor.

Syntax: `[no] ip prefix-list name [seq seq-value] [description string] deny | permit network-addr / mask-bits [ge ge-value] [le le-value]`

The *name* parameter specifies the prefix list name. Use this name when applying the prefix list to a neighbor.

The **description** *string* parameter is a text string describing the prefix list.

The **seq** *seq-value* parameter is optional and specifies the sequence number of the IP prefix list. If you do not specify a sequence number, the software numbers the entries in increments of 5, beginning with prefix list entry 5. The software interprets the prefix list entries in numerical order, beginning with the lowest sequence number.

The **deny** and **permit** parameters specify the action the software takes if a neighbor route is in this prefix list.

The *network-addr* and *mask-bits* parameters specify the network number and the number of bits in the network mask.

You can specify a range of prefix length for prefixes that are more specific than *network-addr* and *mask-bits*.

The prefix-list matches only on this network unless you use the **ge** *ge-value* or **le** *le-value* parameters.

- If you specify only **ge** *ge-value*, the mask-length range is from *ge-value* to 81.
- If you specify only **le** *le-value*, the mask-length range is from length to *le-value*.

The *ge-value* or *le-value* you specify must meet the following condition:

$\text{length} < \text{ge-value} \leq \text{le-value} \leq 81$

If you do not specify **ge** *ge-value* or **le** *le-value*, the prefix list matches only on the exact network prefix you specified with the *network-addr* and *mask-bits* parameters.

In the following example, only default routes are allowed:

```
device(config)# ip prefix-list match-default-routes permit 0.0.0.0/0
```

In the following example, only default routes are denied:

```
device(config)# ip prefix-list match-default-routes deny 0.0.0.0/0
```

In the following example, all routes are allowed, including all subnet masks and all prefixes:

```
device(config)# ip prefix-list match-all-routes permit 0.0.0.0/0 le 32
```

NOTE

Be careful to determine exactly which routes you want to allow using a prefix list.

Defining neighbor distribute lists

A neighbor distribute list is a list of BGP4 address filters or ACLs that filter the traffic to or from a neighbor.

To configure a distribute list that uses ACL 1, enter a command such as the following.

```
device(config-bgp)# neighbor 10.10.10.1 distribute-list 1 in
```

This command configures the device to use ACL 1 to select the routes that the device will accept from neighbor 10.10.10.1.

Syntax: `[no] neighbor ip-addr distribute-list name-or-num in | out`

The *ip-addr* parameter specifies the neighbor.

The *name-or-num* parameter specifies the name or number of a standard or named ACL.

The **in** and **out** parameters specify whether the distribute list applies to inbound or outbound routes:

- **in** - controls the routes the device will accept from the neighbor.
- **out** - controls the routes sent to the neighbor.

Defining 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. If you think of a route map as a table, an instance is a row in that 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** action for routes that match the match clauses:

- If the route map contains a **permit** action, a route that matches a match statement is permitted; otherwise, the route is denied.
- If the route map contains a **deny** action, a route that matches a match statement is denied.
- If a route does not match any match statements in the route map, the route is denied. This is the default action. To change the default action, configure the last match statement in the last instance of the route map to **permit any any**.
- If there is no match statement, the software considers the route to be a match.
- For route maps that contain address filters, AS-path filters, or community filters, if the action specified by a filter conflicts with the action specified by the route map, the route map action takes precedence over the filter action.

If the route map contains set clauses, routes that are permitted by the route map match statements are modified according to the set clauses.

Match statements compare the route against one or more of the following:

- The route BGP4 MED (metric)
- A sequence of AS-path filters
- A sequence of community filters
- A sequence of address filters
- 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 ACL
- A community ACL
- An IP prefix list
- An IP ACL

For routes that match all of the match statements, the route map set clauses 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 and an automatically calculated tag to the route.
- Set the 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.
- Set a BGP4 static network route.

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 clauses, the device sets the attributes in the route according to the set clauses.

To create a route map, you define instances of the map by a sequence number.

To define a route map, use the procedures in the following sections.

Entering the route map into the software

To add instance 1 of a route map named "GET_ONE" with a permit action, enter the following command.

```
device(config)# route-map GET_ONE permit 1
device(config-route-map GET_ONE)#
```

Syntax: [no] route-map *map-name* permit | deny *num*

As shown in this example, the command prompt changes to the route map level. You can enter the match and set clauses at this level.

The *map-name* is a string of characters that names the map. Map names can be up to 80 characters in length.

The **permit** and **deny** parameters specify the action the device will take if a route matches a match statement:

- If you specify **deny**, the device does not advertise or learn the route.
- If you specify **permit**, the device applies the match and set clauses associated with this route map instance.

The *num* parameter specifies the instance of the route map you are defining.

To delete a route map, enter a command such as the following. When you delete a route map, all the permit and deny entries in the route map are deleted.

```
device(config)# no route-map Map1
```

This command deletes a route map named Map1. All entries in the route map are deleted.

To delete a specific instance of a route map without deleting the rest of the route map, enter a command such as the following.

```
device(config)# no route-map Map1 permit 10
```

This command deletes the specified instance from the route map but leaves the other instances of the route map intact.

Specifying the match conditions

Use the following command to define the match conditions for instance 1 of the route map GET_ONE. This instance compares the route updates against BGP4 address filter 11.

```
device(config-route-map GET_ONE)# match address-filters 11
```

Syntax: [no] match [as-path *name*] [community *acl* exact-match] [ip address *acl* | prefix-list *string*] [ip route-source *acl* | prefix *name*] [metric *num*] [next-hop *address-filter-list*] [route-type internal | external-type1 | external-type2] [tag tag-value] [interface *interface interface interface .. protocol bgp static-networkprotocol bgp externalprotocol bgp internal*

The **as-path***num* parameter specifies an AS-path ACL. You can specify up to five AS-path ACLs. To configure an AS-path ACL, use the **ip as-path access-list** command.

The **community** *num* parameter specifies a community ACL.

NOTE

The ACL must already be configured.

The **community***acl***exact-match** parameter matches a route if (and only if) the route community attributes field contains the same community numbers specified in the match statement.

The **ip address**, **next-hop** *acl-num*, **prefix-list**, and *string* parameters specify an ACL or IP prefix list. Use this parameter to match based on the destination network or next-hop gateway. To configure an IP ACL for use with this command, use the **ip access-list** command. To configure an IP prefix list, use the **ip prefix-list** command.

The **ip route-source***acl* and **prefix***name* parameters match based on the source of a route (the IP address of the neighbor from which the device learned the route).

The **metric***num* parameter compares the route MED (metric) to the specified value.

The **next-hop** *address-filter-list* parameter compares the IP address of the route next-hop to the specified IP address filters. The filters must already be configured.

The **route-type** *internal*, **external-type1**, and **external-type2** parameters apply only to OSPF routes. These parameters compare the route type to the specified value.

The **tag***tag-value* parameter compares the route tag to the specified tag value.

The *protocol* **bgp static-network** parameter matches on BGP4 static network routes.

The *protocol* **bgp external** parameter matches on eBGP (external) routes.

The *protocol* **bgp internal** parameter matches on iBGP (internal) routes.

Match examples using ACLs

The following sections contain examples of how to configure route maps that include match statements that match on ACLs.

Matching based on AS-path ACL

To construct a route map that matches based on AS-path ACL 1, enter the following commands.

```
device(config)# route-map PathMap permit 1
device(config-route-map PathMap)# match as-path 1
```

Syntax: **[no] match as-path** *string*

The *string* parameter specifies an AS-path ACL and can be a number from 1 through 199. You can specify up to five AS-path ACLs.

Matching based on community ACL

To construct a route map that matches based on community ACL 1, enter the following commands.

```
device(config)# ip community-list 1 permit 123:2
device(config)# route-map CommMap permit 1
device(config-route-map CommMap)# match community 1
```

Syntax: **[no] match community** *string*

The *string* parameter specifies a community list ACL. To configure a community list ACL, use the **ip community-list** command.

Matching based on destination network

You can use the results of an IP ACL or an IP prefix list as the match condition.

To construct a route map that matches based on destination network, enter commands such as the following.

```
device(config)# route-map NetMap permit 1
device(config-routemap NetMap)# match ip address 1
```

Syntax: [no] match ip address *ACL-name-or-num*

Syntax: [no] match ip address prefix-list *name*

The *ACL-name-or-num* parameter with the first command specifies an IP ACL and can be a number from 1 through 199 or the ACL name if it is a named ACL. Multiple ACLs may be added when separated by spaces. To configure an IP ACL, use the **ip access-list** or **access-list** command.

The *name* parameter with the second command specifies an IP prefix list name.

Matching based on next-hop device

You can use the results of an IP ACL or an IP prefix list as the match condition.

To construct a route map that matches based on the next-hop device, enter commands such as the following.

```
device(config)# route-map HopMap permit 1
device(config-routemap HopMap)# match ip next-hop 2
```

Syntax: [no] match ip next-hop *string*

Syntax: [no] match ip next-hop prefix-list *name*

The *string* parameter with the first command specifies an IP ACL and can be a number from 1 through 199 or the ACL name if it is a named ACL. To configure an IP ACL, use the **ip access-list** or **access-list** command.

The *name* parameter with the second command specifies an IP prefix list name.

Matching based on the route source

To match a BGP4 route based on its source, use the **match ip route-source** command.

```
device(config)# access-list 10 permit 192.168.6.0 0.0.0.255
device(config)# route-map bgp1 permit 1
device(config-routemap bgp1)# match ip route-source 10
```

The first command configures an IP ACL that matches on routes received from 192.168.6.0/24. The remaining commands configure a route map that matches on all BGP4 routes advertised by the BGP4 neighbors whose addresses match addresses in the IP prefix list.

You can add a set clause to change a route attribute in the routes that match. You also can use the route map as input for other commands, such as the **neighbor** and **network** commands and some show commands.

Syntax: [no] match ip route-source *ACL* | prefix-list *name*

The *acl* and **prefix-list** *name* parameters specify the name or ID of an IP ACL, or an IP prefix list.

Matching on routes containing a specific set of communities

The device can match routes based on the presence of a community name or number in a route. To match based on a set of communities, configure a community ACL that lists the communities, then compare routes against the ACL.

```
device(config)# ip community-list standard std_1 permit 12:34 no-export
device(config)# route-map bgp2 permit 1
device(config-routemap bgp2)# match community std_1 exact-match
```

The first command configures a community ACL that contains community number 12:34 and community name no-export. The remaining commands configure a route map that matches the community attributes field in BGP4 routes against the set of communities in the ACL. A route matches the route map only if the route contains all the communities in the ACL and no other communities.

Syntax: [no] match community *ACL* exact-match

The *ACL* parameter specifies the name of a community list ACL. You can specify up to five ACLs. Separate the ACL names or IDs with spaces.

Here is another example.

```
device(config)# ip community-list standard std_2 permit 23:45 56:78
device(config)# route-map bgp3 permit 1
device(config-routemap bgp3)# match community std_1 std_2 exact-match
```

These commands configure an additional community ACL, *std_2*, that contains community numbers 23:45 and 57:68. Route map *bgp3* compares each BGP4 route against the sets of communities in ACLs *std_1* and *std_2*. A BGP4 route that contains either but not both sets of communities matches the route map. For example, a route containing communities 23:45 and 57:68 matches. However, a route containing communities 23:45, 57:68 and 12:34, or communities 23:45, 57:68, 12:34, and no-export does not match. To match, the route communities must be the same as those in exactly one of the community ACLs used by the match community statement.

Matching based on BGP4 static network

The **match** option has been added to the **route-map** command that allows you to match on a BGP4 static network. In the following example, the route-map is configured to match on the BGP4 static network. The device is then configured to advertise to the core BGP4 peer (IP address 192.168.6.0) only the BGP4 static routes and nothing else.

```
device(config)# route-map policygroup3 permit 10
device(config-routemap policygroup3)# match protocol bgp static-network
device(config-routemap policygroup3)# set local-preference 150
device(config-routemap policygroup3)# set community no-export
device(config-routemap policygroup3)# exit
device(config)# router bgp
device(config-bgp)# neighbor 192.168.6.0 route-map out policymap3
```

Syntax: [no] match protocol bgp [external | internal | static-network]

The **match protocol bgp external** option will match the eBGP routes.

The **match protocol bgp internal** option will match the iBGP routes.

The **match protocol bgp static-network** option will match the static-network BGP4 route, applicable at BGP4 outbound policy only.

Matching based on interface

The **match** option has been added to the **route-map** command that distributes any routes that have their next hop out one of the interfaces specified. This feature operates with the following conditions:

- The **match interface** option can only use the interface name (for example ethernet 1/1/2) and not the IP address as an argument.

- The **match interface** option is only effective during redistribution and does not apply for other route map usage such as: bgp outbound route update policy.
- The **match interface** option can be applied to other types of redistribution such as redistributing OSPF routes to BGP4, or filtering out all OSPF routes that point to a specific interface.

To configure the match-interface option, use the following command.

```
device(config)# route-map test-route permit 99
device(config-routemap test-route)# match interface ethernet 1/1/1 ethernet 1/3/2
device(config-routemap test-route)# exit
```

Syntax: [no] **match interface** *interface interface* ...

The *interface* variable specifies the interface that you want to use with the **match interface** command. Up to 5 interfaces of the following types can be specified:

- **ethernet** *unit/slot/port*
- **loopback** *loopback-number*
- **null0**
- **tunnel** *tunnel-ID*
- **ve** *ve-ID*

Setting parameters in the routes

Use the following command to define a set clause that prepends an AS number to the AS path on each route that matches the corresponding match statement.

```
device(config-routemap GET_ONE)# set as-path prepend 65535
```

Syntax: [no] **set** [**as-path** [**prepend** *as-num,as-num,...*]] [**automatic-tag**] [**comm-list** *acl delete*] [**community** *num : num | num | additive | local-as | no-advertise | no-export*] [**dampening** [*half-life reuse suppress max-suppress-time*]] [**ip next hop** *ip-addr*] [**ip next-hop peer-address**] [**local-preference** *num*] [**metric** [*+ | -*] *num* | **none**] [**metric-type** *type-1 | type-2*] [**external** [**metric-type** *internal*]] [**next-hop** *ip-addr*] [**origin** *igp | incomplete*] [**tag**] [**weight** *num*]

The **as-path prepend***num,num,...* parameter adds the specified AS numbers to the front of the AS-path list for the route. The range of num values is 1 - 65535 for two-byte ASNs and 1 - 4294967295 if AS4s have been enabled.

The **automatic-tag** parameter calculates and sets an automatic tag value for the route.

NOTE

This parameter applies only to routes redistributed into OSPF.

The **comm-list** parameter deletes a community from the community attributes field for a BGP4 route.

The **community** parameter sets the community attribute for the route to the number or well-known type you specify.

The **dampening** [*half-life reuse suppress max-suppress-time*] parameter sets route dampening parameters for the route. The *half-life* parameter specifies the number of minutes after which the route penalty becomes half its value. The *reuse* parameter specifies how low a route penalty must become before the route becomes eligible for use again after being suppressed. The *suppress* parameter specifies how high a route penalty can become before the device suppresses the route. The *max-suppress-time* parameter specifies the maximum number of minutes that a route can be suppressed regardless of how unstable it is.

The **ip next hop** *ip-addr* parameter sets the next-hop IP address for route that matches a match statement in the route map.

The **ip next-hop peer-address** parameter sets the BGP4 next hop for a route to the neighbor address.

The **local-preference** *num* parameter sets the local preference for the route. You can set the preference to a value from 0 through 4294967295.

The **metric** **[+ | -]** *num* **| none** parameter sets the MED (metric) value for the route. The default MED value is 0. You can set the preference to a value from 0 through 4294967295.

- **set metric** *num* - Sets the metric for the route to the number you specify.
- **set metric +** *num* - Increases route metric by the number you specify.
- **set metric -** *num* - Decreases route metric by the number you specify.
- **set metric none** - Removes the metric from the route (removes the MED attribute from the BGP4 route).

The **metric-type** **type-1** and **type-2** parameters change the metric type of a route redistributed into OSPF.

The **metric-type** **internal** parameter sets the route MED to the same value as the IGP metric of the BGP4 next-hop route. The parameter does this when advertising a BGP4 route to an EBGp neighbor.

The **next-hop** *ip-addr* parameter sets the IP address of the route next-hop device.

The **origin** **igp** **incomplete** parameter sets the route origin to IGP or INCOMPLETE.

The **tag***tag-value* parameter sets the route tag. You can specify a tag value from 0 through 4294967295.

NOTE

This parameter applies only to routes redistributed into OSPF.

NOTE

You also can set the tag value using a table map. The table map changes the value only when the device places the route in the IP route table instead of changing the value in the BGP4 route table.

The **weight** *num* parameter sets the weight for the route. The range for the weight value is 0 through 4294967295.

Setting a BGP4 route MED to equal the next-hop route IGP metric

To set a route's MED to the same value as the IGP metric of the BGP4 next-hop route, when advertising the route to a neighbor, enter commands such as the following.

```
device(config)# access-list 1 permit 192.168.9.0 0.0.0.255
device(config)# route-map bgp4 permit 1
device(config-route-map bgp4)# match ip address 1
device(config-route-map bgp4)# set metric-type internal
```

The first command configures an ACL that matches on routes with destination network 192.168.9.0. The remaining commands configure a route map that matches on the destination network in ACL 1, then sets the metric type for those routes to the same value as the IGP metric of the BGP4 next-hop route.

Syntax: no set metric-type internal

Setting the next-hop of a BGP4 route

To set the next-hop address of a BGP4 route to a neighbor address, enter commands such as the following.

```
device(config)# route-map bgp5 permit 1
device(config-route-map bgp5)# match ip address 1
device(config-route-map bgp5)# set ip next-hop peer-address
```

These commands configure a route map that matches on routes whose destination network is specified in ACL 1, and sets the next hop in the routes to the neighbor address (inbound filtering) or the local IP address of the BGP4 session (outbound filtering).

Syntax: [no] set ip next-hop peer-address

The value that the software substitutes for **peer-address** depends on whether the route map is used for inbound filtering or outbound filtering:

- When you use the **set ip next-hop peer-address** command in an inbound route map filter, **peer-address** substitutes for the neighbor IP address.
- When you use the **set ip next-hop peer-address** command in an outbound route map filter, **peer-address** substitutes for the local IP address of the BGP4 session.

NOTE

You can use this command for a peer group configuration.

Deleting a community from a BGP4 route

To delete a community from a BGP4 route's community attributes field, enter commands such as the following.

```
device(config)# ip community-list standard std_3 permit 12:99 12:86
device(config)# route-map bgp6 permit 1
device(config-routemap bgp6)# match ip address 1
device(config-routemap bgp6)# set comm-list std_3 delete
```

The first command configures a community ACL containing community numbers 12:99 and 12:86. The remaining commands configure a route map that matches on routes whose destination network is specified in ACL 1, and deletes communities 12:99 and 12:86 from those routes. The route does not need to contain all the specified communities in order for them to be deleted. For example, if a route contains communities 12:86, 33:44, and 66:77, community 12:86 is deleted.

Syntax: [no] set comm-list ACL delete

The **ACL** parameter specifies the name of a community list ACL.

Using a table map to set the tag value

Route maps that contain set statements change values in routes when the routes are accepted by the route map. For inbound route maps (route maps that filter routes received from neighbors), the routes are changed before they enter the BGP4 route table.

For tag values, if you do not want the value to change until a route enters the IP route table, you can use a table map to change the value. A table map is a route map that you have associated with the IP routing table. The device applies the set statements for tag values in the table map to routes before adding them to the route table.

To configure a table map, you first configure the route map, then identify it as a table map. The table map does not require separate configuration. You can have one table map.

NOTE

Use table maps only for setting the tag value. Do not use table maps to set other attributes. To set other route attributes, use route maps or filters.

To create a route map and identify it as a table map, enter commands such as following. These commands create a route map that uses an address filter. For routes that match the IP prefix list filter, the route map changes the tag value to 100 and is then considered as a table map. This route map is applied only to routes the device places in the IP route table. The route map is not applied to all routes. This example assumes that IP prefix list p11 has already been configured.

```
device(config)# route-map TAG_IP permit 1
device(config-routemap TAG_IP)# match ip address prefix-list p11
```



```
device(config-routemap TAG_IP)# set tag 100
device(config-routemap TAG_IP)# router bgp

device(config-bgp)# table-map TAG_IP
```

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.

Enabling cooperative filtering

To configure cooperative filtering, enter commands such as the following.

```
device(config)# ip prefix-list Routesfrom10234 deny 10.20.0.0/24
device(config)# ip prefix-list Routesfrom10234 permit 0.0.0.0/0 le 32
device(config)# router bgp
device(config-bgp-router)# neighbor 10.2.3.4 prefix-list Routesfrom1234 in
device(config-bgp-router)# neighbor 10.2.3.4 capability orf prefixlist send
```

The first two commands configure statements for the IP prefix list Routesfrom1234. The first command configures a statement that denies routes to 10.20.0.0/24. The second command configures a statement that permits all other routes. Once you configure an IP prefix list statement, all routes not explicitly permitted by statements in the prefix list are denied.

The next two commands change the CLI to the BGP4 configuration level, then apply the IP prefix list to neighbor 10.2.3.4. The last command enables the device to send the IP prefix list as an ORF to neighbor 10.2.3.4. When the device sends the IP prefix list to the neighbor, the neighbor filters out the 10.20.0.x routes from its updates to the device. This assumes that the neighbor is also configured for cooperative filtering.

Syntax: `[no] neighbor ip-addr | peer-group-name capability orf prefixlist [send | receive]`

The `ip-addr | peer-group-name` parameters specify the IP address of a neighbor or the name of a peer group of neighbors.

The **send** and **receive** parameters specify the support you are enabling:

- **send** - The device sends the IP prefix lists to the neighbor.
- **receive** - The device accepts filters from the neighbor.

If you do not specify the capability, both capabilities are enabled.

The **prefixlist** parameter specifies the type of filter you want to send to the neighbor.

NOTE

The current release supports cooperative filtering only for filters configured using IP prefix lists.

Sending and receiving ORFs

Cooperative filtering affects neighbor sessions that start after the filtering is enabled, but do not affect sessions that are already established.

To activate cooperative filtering, reset the session with the neighbor. This is required because the cooperative filtering information is exchanged in Open messages during the start of a session.

To place a prefix-list change into effect after activating cooperative filtering, perform a soft reset of the neighbor session. A soft reset does not end the current session, but sends the prefix list to the neighbor in the next route refresh message.

NOTE

Make sure cooperative filtering is enabled on the device and on the neighbor before you send the filters.

To reset a neighbor session and send ORFs to the neighbor, enter a command such as the following.

```
device# clear ip bgp neighbor 10.2.3.4
```

This command resets the BGP4 session with neighbor 10.2.3.4 and sends the ORFs to the neighbor. If the neighbor sends ORFs to the device, the device accepts them if the send capability is enabled.

To perform a soft reset of a neighbor session and send ORFs to the neighbor, enter a command such as the following.

```
device# clear ip bgp neighbor 10.2.3.4 soft in prefix-list
```

Syntax: `clear ip bgp neighbor ip-addr [soft in prefix-filter | soft in prefix-list]`

If you use the **soft in prefix-filter** parameter, the device sends the updated IP prefix list to the neighbor as part of its route refresh message to the neighbor.

NOTE

If the device or the neighbor is not configured for cooperative filtering, the command sends a normal route refresh message.

Displaying cooperative filtering information

You can display the following cooperative filtering information:

- The cooperative filtering configuration on the device.
- The ORFs received from neighbors.

To display the cooperative filtering configuration on the device, enter a command such as the following.

```
device# show ip bgp neighbor 10.10.10.1
1 IP Address: 10.10.10.1, AS: 65200 (IBGP), RouterID: 10.10.10.1
  State: ESTABLISHED, Time: 0h0m7s, KeepAliveTime: 60, HoldTime: 180
  RefreshCapability: Received
  CooperativeFilteringCapability: Received
Messages:   Open      Update  KeepAlive  Notification  Refresh-Req
Sent       : 1        0        1          0              1
Received: 1        0        1          0              1
Last Update Time: NLRI      Withdraw      NLRI      Withdraw
                  Tx: ---      ---          Rx: ---      ---
Last Connection Reset Reason:Unknown
Notification Sent:      Unspecified
Notification Received: Unspecified
TCP Connection state: ESTABLISHED
Byte Sent: 110, Received: 110
Local host: 10.10.10.2, Local Port: 8138
Remote host: 10.10.10.1, Remote Port: 179
ISentSeq: 460 SendNext: 571 TotUnAck: 0
TotSent: 111 ReTrans: 0 UnAckSeq: 571
IRcvSeq: 7349 RcvNext: 7460 SendWnd: 16384
TotalRcv: 111 DupliRcv: 0 RcvWnd: 16384
SendQue: 0 RcvQue: 0 CngstWnd: 5325
```

Syntax: show ip bgp neighbor *ip-addr*

To display the ORFs received from a neighbor, enter a command such as the following:

```
device# show ip bgp neighbor 10.10.10.1 received prefix-filter
ip prefix-list 10.10.10.1: 4 entries
seq 5 permit 10.10.0.0/16 ge 18 le 28
seq 10 permit 10.20.10.0/24
seq 15 permit 10.0.0.0/8 le 32
seq 20 permit 10.10.0.0/16 ge 18
```

Syntax: show ip bgp neighbor *ip-addr* received prefix-filter

Four-byte Autonomous System Numbers (AS4)

This section describes the reasons for enabling four-byte autonomous system numbers (AS4s). AS4s are supported by default. You can specify and view AS4s by default and using the enable facility described in this section. However, not all devices in a network are always capable of utilizing AS4s. The act of enabling them on the local device initiates a facility for announcing the capability and negotiating its use with neighbors. If you do not enable AS4s on a device, other devices do not know that this device is sending them.

The system uses a hierarchy to prioritize the utilization of the AS4 capability. The prioritization depends on the CLI configuration commands. AS4s can be enabled and configured at the level of a neighbor, a peer group, or globally for the entire device, according to the following bottom-up hierarchy:

- If a neighbor has no configuration for AS4s but it belongs to a peer group, the neighbor uses the configuration from the peer group. For example, if you configure a neighbor but do not include a specification for AS4s, one of the following applies:
 - The neighbor uses the AS4 configuration for a peer group if it belongs to a peer group.
 - The neighbor uses the device configuration if it does not belong to a peer group or the peer group has no AS4 configuration.
- If a peer group has no configuration for AS4s, it can use the global configuration of the device. If the device has no configuration for AS4s, then a neighbor or peer group without a configuration for AS4s use the device default--no announcement or negotiation of AS4s.

- If a neighbor belongs to peer group with an AS4 configuration but you want that neighbor to be disabled or have a different AS4 configuration, the neighbor AS4 configuration overrides the peer group configuration. For example, you can ensure that neighbor has no AS4 announcement and negotiation activity even though the peer group is enabled for AS4 capability.

NOTE

The configuration for AS4 can be enabled, disabled, or can have no explicit configuration.

CLI commands allow you to disable AS4s on an entity whose larger context has AS4s enabled. For example, you can use a CLI command to disable AS4s on a neighbor that is a member of a peer group that is enabled for AS4s.

Normally, AS4s are sent only to a device, peer group, or neighbor that is similarly configured for AS4s. If a AS4 is configured for a local-autonomous systemS, the system signals this configuration by sending AS_TRANS in the My Autonomous System field of the OPEN message. However, if the AS4 capability for a neighbor is disabled, the local device does not send the four-byte Autonomous System number capability to the neighbor.

Enabling AS4 numbers

This section describes how to enable the announcement and negotiation of AS4s and describes the different types of notation that you can use to represent a AS4.

You can enable AS4s on a device, a peer group, and a neighbor. For global configuration, the **capability** command in the BGP4 configuration context enables or disables AS4 support. For a peer group or a neighbor, **capability** is a keyword for the **neighbor** command. In addition to enabling AS4s for a neighbor or a peer group, you can also use the combination of the **capability** keyword and the optional **enable** or **disable** keyword to disable this feature in a specific case where the AS4s are enabled for a larger context. The Neighbor configuration of AS4s section illustrates this capability.

Global AS4 configuration

To enable AS4s globally, use the **capability** command in the BGP4 configuration context as shown.

```
device(config-bgp)# capability as4 enable
```

Syntax: [no] **capability as4 enable | disable**

The **no** form of the **capability** command deletes the announcement and negotiation configuration of AS4s (if it has been enabled) at the global level. Using the regular form of the command with the **disable** keyword has the same effect on the global configuration. Disabling or using the **no** form of the command does not affect the configuration at the level of a peer or neighbor.

The consequences of choosing between the **enable** or **disable** keyword are reflected in the output of the **show running configuration** command.

Peer group configuration of AS4s

To enable AS4s for a peer group, use the **capability** keyword with the **neighbor** command in the BGP4 configuration context, as the following example for the Peergroup_1 peer group illustrates.

```
device(config-bgp)# neighbor Peergroup_1 capability as4 enable
```

Syntax: [no] **neighbor peer-group-name capability as4 enable | disable**

The **no** form of the **neighbor** command along with the **capability** and **as4** keywords disables the announcement and negotiation of AS4s in the named peer group. Using the regular form of the command with the **disable** keyword has the same effect on the neighbor configuration.

The consequences using the **enable** or **disable** keywords are reflected in the output of the **show running configuration** command. However, if the peer group configuration omits an explicit AS4 argument, the **show running configuration** output will not contain AS4 information.

Neighbor configuration of AS4s

To enable AS4s for a neighbor, use the **capability** and **as4** keywords with the **neighbor** command in the BGP4 configuration context, as the following example for IP address 1.1.1.1 illustrates.

```
device(config-bgp) # neighbor 1.1.1.1 capability as4 enable
```

Syntax: [no] neighbor IPaddress capability as4 enable | disable

The **no** form of the **neighbor** command with the **capability** and **as4** keywords deletes the neighbor-enable for AS4s.

The consequences of using the **enable** or **disable** keywords are reflected in the output of the **show running configuration** command. However, if the neighbor configuration omits an explicit AS4 argument, the **show running configuration** output will not contain AS4 information.

To disable AS4s on a particular neighbor within a peer group that is enabled for AS4s, enter a command similar to the following.

```
device(config-bgp) # neighbor 1.1.1.1 capability as4 disable
```

Specifying the local AS number

The local autonomous system number (ASN) identifies the autonomous system where the BGP4 device resides.

Normally, AS4s are sent only to a device, peer group, or neighbor that is similarly configured for AS4s. Typically, if you try to set up a connection from an AS4-enabled device to a device that processes only two-byte ASNs, the connection fails to come up unless you specify the reserved ASN 23456 as the local ASN to send to the far-end device.

To set the local autonomous system number, enter commands such as the following.

```
device(config) # router bgp
BGP4: Please configure 'local-as' parameter in order to enable BGP4.
device(config-bgp) # local-as 100000
device(config-bgp) # write memory
```

Syntax: [no] local-as num

The *num* parameter specifies a local ASN in the range 1 - 4294967295. No default exists for *num*. ASNs 64512 - 65535 are the well-known private BGP4 autonomous system numbers and are not advertised to the Internet community.

Route-map set commands and AS4s

You can prepend an AS4 number to an autonomous system path or make the autonomous system number a tag attribute for a route map as shown here.

```
device(config-routemap test) # set as-path prepend 7701000
```

Syntax: [no] set as-path prepend num,num , ... | tag

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

NOTE

If the autonomous system path for a route map has prepended ASNs and you want to use the **no** form of the command to delete the configuration, you must include the prepended ASNs in the **no set as-path** entry. For example, if 70000 and 70001 have been prepended to a route map, enter **no set as-path prepend 70000 70001**. As a shortcut, in the configuration context of a particular route map, you can also copy and paste ASNs from the output of **show** commands, such as **show route-map** or **show ip bgp route**.

Use the **prepend** keyword to prepend one or more ASNs. The maximum number of ASNs that you can prepend is 16. The range for each ASN is 1 - 4294967295.

Entering the **tag** keyword sets the tag as an AS-path attribute.

Clearing BGP4 routes to neighbors

You can clear BGP4 connections using the AS4 as an argument with the **clear ip bgp neighbor** command in the configuration context level of the CLI, as shown.

```
device(config)# clear ip bgp neighbor 80000
```

Syntax: **clear ip bgp neighbor** *all* | *ip-addr* | *peer-group-name* | *as-num* [*last-packet-with-error* | *notification-errors*] [*soft* [*in* | *out*] | *soft-outbound*]

The neighbor specification is either *all*, *ip-addr*, *peer-group-name*, or *as-num*. The **all** parameter specifies all neighbors. The *ip-addr* parameter specifies a neighbor by its IP interface with the device. The *peer-group-name* specifies all neighbors in a specific peer group. The *as-num* parameter specifies all neighbors within the specified AS. After choosing one mandatory parameter, you can choose an optional parameter.

The **soft in** and **soft out** parameters determine whether to refresh the routes received from the neighbor or the routes sent to the neighbor. If you do not specify **in** or **out**, the device performs a soft refresh in both options:

- **soft in** performs one of the following actions on inbound routes, according to other configuration settings:
 - If you enabled soft reconfiguration for the neighbor or peer group, **soft in** updates the routes by comparing the route policies against the route updates that the device has stored. Soft reconfiguration does not request additional updates from the neighbor or otherwise affect the session with the neighbor.
 - If you did not enable soft reconfiguration, **soft in** requests the entire BGP4 route table on the neighbor (Adj-RIB-Out), then applies the filters to add, change, or exclude routes.
 - If a neighbor does not support dynamic refresh, **soft in** resets the neighbor session.
- **soft out** updates all outbound routes and then sends the entire BGP4 route table for the device (Adj-RIB-Out) to the neighbor after the device changes or excludes the routes affected by the filters.
- The **soft-outbound** parameter updates all outbound routes by applying the new or changed filters, but sends only the existing routes affected by the new or changed filters to the neighbor.

NOTE

Use **soft-outbound** only if the outbound policy is changed. The **soft-outbound** parameter updates all outbound routes by applying the new or changed filters. However, the device sends to the neighbor only the existing routes that are affected by the new or changed filters. The **soft out** parameter updates all outbound routes and then sends the entire BGP4 route table on the device to the neighbor after the device changes or excludes the routes affected by the filters.

AS4 notation

A AS4 can appear in either a plain or a dot notation format in the output of **show** commands. To select one of these formats, specify the format before entering the **show** command. This section defines these formats and describes how to select a format. The following notations are currently supported:

- With the default **asplain**, the ASN is a decimal integer in the range 1 - 4294967295.
- With **asdot** +, all ASNs are two integer values joined by a period character in the following format:

<high order 16-bit value in decimal>.<low order 16-bit value in decimal>

Using the **asdot** notation, an autonomous system number of value 65526 is represented as the string "0.65526," and an autonomous system number of value 65546 is represented as the string "1.10."

- With **asdot**, an ASN less than 65536 uses the **asplain** notation (and represents autonomous system number values equal to or greater than 65536 using the **asdot** notation). Using the **asdot** notation, ASN 65526 is represented as the string "65526," and ASN 65546 is represented as the string "1.10".

NOTE

You can enter autonomous system numbers in any format. However, if you want the **asdot** or the **asdot** + format to appear in the output of a **show** command, you must specify these in the CLI.

NOTE

Remember that autonomous system path matching that uses regular expression is based on the configured autonomous system format.

The following command sequences show how to enable the different notations for AS4s and how these notations appear in the output display.

To see ASNs in **asplain**, use the **show ip bgp** command.

```
device(config)# show ip bgp
Total number of BGP Routes: 1
Status codes: s suppressed, d damped, h history, * valid, > best, i internal, S stale
Origin codes: i - IGP, e - EGP, ? - incomplete
Network      Next Hop      Metric LocPrf Weight Path
*> 47.1.1.0/24 192.168.1.5    1      100    0      90000 100 200 65535 65536 65537 65538 65539 75000 ?
```

To specify **asdot** notation before displaying IP BGP4 information, use the **as-format** command.

```
device(config)# as-format asdot
device(config)# show ip bgp
Total number of BGP Routes: 1
Status codes: s suppressed, d damped, h history, * valid, > best, i internal, S stale
Origin codes: i - IGP, e - EGP, ? - incomplete
Network      Next Hop      Metric LocPrf Weight Path
*> 10.1.1.0/24 192.168.1.5    1      100    0      1.24464 100 200 65535
1.0 1.1 1.2 1.3 1.9464 ?
```

Syntax: [no] **as-format asplain | asdot | asdot+**

The default is **asplain** and can be restored using the **no** version of the command, if the CLI is currently using **asdot** or **asdot+**.

To activate **asdot+** notation, enter **as-format asdot+** in the CLI.

```
device(config)# as-format asdot+
device(config)# show ip bgp
Total number of BGP Routes: 1
Status codes: s suppressed, d damped, h history, * valid, > best, i internal, S stale
Origin codes: i - IGP, e - EGP, ? - incomplete
Network      Next Hop      Metric LocPrf Weight Path
*> 10.1.1.0/24 192.168.1.5    1      100    0      1.24464 0.100 0.200
0.65535 1.0 1.1 1.2 1.3 1.9464 ?
```

BGP4 AS4 attribute errors

This section describes the handling of the confederation path segments in the AS4_PATH attribute, and also specifies the error handling for the new attributes.

To support AS4, the following attributes: AS4_PATH and AS4_Aggregator were specified in RFC 4893. Confederation path segments in an AS4_PATH are discarded and if there are any other errors such as: *attribute length* , *flag* , confederation segments after AS_SEQ/AS_SET, Invalid segment types and More than one AS4_PATH in these new attributes, the attribute is discarded and the error is logged.

Error logs

The device generates a log when it encounters attribute errors in AS4_PATH and AS4_AGGREGATOR.

NOTE

Logging of errors is rate-limited to not more than one message for every two minutes. Some errors may be lost due to this rate-limiting.

Sample log messages for various attribute errors are shown here.

Attribute length error (ignore the AS4_PATH)

```
SYSLOG: Sep  9 19:02:03:<11>mu2, BGP: From Peer 192.168.1.1 received invalid AS4_PATH attribute length (3)
- entire AS4_PATH ignored
```

Attribute flag error (ignore the AS4_PATH)

```
SYSLOG: Sep  9 19:02:03:<11>mu2, BGP: From Peer 192.168.1.1 received invalid AS4_PATH attribute flag (0x40)
- entire AS4_PATH ignored
```

Confederation segments after AS_SEQ/AS_SET (ignore the AS4_PATH)

```
SYSLOG: Sep  9 19:02:03:<11>mu2, BGP: From Peer 192.168.1.1 received invalid Confed info in AS4_PATH (@byte
43) - entire AS4_PATH not ignored
```

Invalid segment types (ignore the AS4_PATH)

```
SYSLOG: Sep  9 19:02:03:<11>mu2, BGP: From Peer 192.168.1.1 received incorrect Seq type/len in AS4_PATH
(@byte 41) - entire AS4_PATH ignored
```

More than one AS4_PATH (Use the first one and ignore the others)

```
SYSLOG: Sep  9 19:02:03:<11>mu2, BGP: From Peer 192.168.1.1 received multiple AS4_PATH attributes - used
first AS4_PATH attribute only
```

Configuring 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 Brocade implementation of route flap dampening is based on RFC 2439.

Route flap dampening is disabled by default. You can enable the feature globally or on an individual route basis using route maps.

NOTE

The device applies route flap dampening only to routes learned from EBGp neighbors.

The route flap dampening mechanism is based on penalties. When a route exceeds a configured penalty value, the device stops using that route and stops advertising it to other devices. The mechanism also allows route penalties to reduce over time if route stability improves.

The route flap dampening mechanism uses the following parameters:

- **Suppression threshold** - Specifies the penalty value at which the device stops using the route. Each time a route becomes unreachable or is withdrawn by a BGP4 UPDATE from a neighbor, the route receives a penalty of 1000. By default, when a route penalty is greater than 2000, the device stops using the route. By default, if a route goes down more than twice, the device stops using the route. You can set the suppression threshold to a value from 1 through 20000. The default is 2000.
- **Half-life** - Once a route has been assigned a penalty, the penalty decreases exponentially and decreases by half after the half-life period. The default half-life period is 15 minutes. The software reduces route penalties every five seconds. For example, if a route has a penalty of 2000 and does not receive any more penalties during the half-life, the penalty is reduced to 1000 after the half-life expires. You can configure the half-life to be from 1 - 45 minutes. The default is 15 minutes.
- **Reuse threshold** - Specifies the minimum penalty a route can have and still be suppressed by the device. If the route penalty falls below this value, the device un-suppresses the route and can use it again. The software evaluates the dampened routes every ten seconds and un-suppresses the routes that have penalties below the reuse threshold. You can set the reuse threshold to a value from 1 through 20000. The default is 750.
- **Maximum suppression time** - Specifies the maximum number of minutes a route can be suppressed regardless of how unstable the route has been before this time. You can set the parameter to a value from 1 through 20000 minutes. The default is four times the half-life. When the half-life value is set to its default (15 minutes), the maximum suppression time defaults to 60 minutes.

You can configure route flap dampening globally or for individual routes using route maps. If you configure route flap dampening parameters globally and also use route maps, the settings in the route maps override the global values.

Globally configuring route flap dampening

Route flap dampening reduces the amount of route state changes propagated by BGP4 due to unstable routes. This in turn reduces processing requirements.

To enable route flap dampening using the default values, enter the following command.

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

Syntax: `[no] dampening [half-life reuse suppress max-suppress-time]`

The *half-life* parameter specifies the number of minutes after which the penalty for a route becomes half its value. The route penalty allows routes that have remained stable for a period despite earlier instability to eventually become eligible for use again. The decay rate of the penalty is proportional to the value of the penalty. After the half-life expires, the penalty decays to half its value. A dampened route that is no longer unstable can eventually again become eligible for use. You can configure the half-life to be from 1 through 45 minutes. The default is 15 minutes.

The *reuse* parameter specifies how low a penalty for a route must be before the route becomes eligible for use again, after being suppressed. You can set the reuse threshold to a value from 1 through 20000. The default is 750 (0.75, or three-fourths, of the penalty assessed for a one flap).

The *suppress* parameter specifies how high the penalty for a route can be before the device suppresses the route. You can set the suppression threshold to a value from 1 through 20000. The default is 2000 (more than two flaps).

The *max-suppress-time* parameter specifies the maximum number of minutes that a route can be suppressed regardless of how unstable it is. You can set the maximum suppression time to a value from 1 through 255 minutes. The default is 40 minutes.

This example shows how to change the dampening parameters.

```
device(config-bgp-router)# dampening 20 200 2500 40
```

This command changes the half-life to 20 minutes, the reuse threshold to 200, the suppression threshold to 2500, and the maximum number of minutes a route can be dampened to 40.

NOTE

To change any of the parameters, you must specify all the parameters with the command. To want to leave any parameters unchanged, enter their default values.

Using a route map to configure route flap dampening for a specific neighbor

You can use a route map to configure route flap dampening for a specific neighbor by performing the following tasks:

- Configure an empty route map with no match or set clauses. This route map does not specify particular routes for dampening but does allow you to enable dampening globally when you refer to this route map from within the BGP4 configuration level.
- Configure another route map that explicitly enables dampening. Use a set clause within the route map to enable dampening. When you associate this route map with a specific neighbor, the route map enables dampening for all routes associated with the neighbor. You also can use match clauses within the route map to selectively perform dampening on some routes from the neighbor.

NOTE

You still need to configure the first route map to enable dampening globally. The second route map does not enable dampening by itself; it just applies dampening to a neighbor.

- Apply the route map to the neighbor.

To enable route flap dampening for a specific BGP4 neighbor, enter commands such as the following.

```
device(config)# route-map DAMPENING_MAP_ENABLE permit 1
device(config-routemap DAMPENING_MAP_ENABLE)# exit
device(config)# route-map DAMPENING_MAP_NEIGHBOR_A permit 1
device(config-routemap DAMPENING_MAP_NEIGHBOR_A)# set dampening
device(config-routemap DAMPENING_MAP_NEIGHBOR_A)# exit
device(config)# router bgp
device(config-bgp)# dampening route-map DAMPENING_MAP_ENABLE
device(config-bgp)# neighbor 10.10.10.1 route-map in DAMPENING_MAP_NEIGHBOR_A
```

In this example, the first command globally enables route flap dampening. This route map does not contain any match or set clauses. At the BGP4 configuration level, the **dampening route-map** command refers to the DAMPENING_MAP_ENABLE route map created by the first command, thus enabling dampening globally.

The third and fourth commands configure a second route map that explicitly enables dampening. Notice that the route map does not contain a match clause. The route map implicitly applies to all routes. Since the route map will be applied to a neighbor at the BGP4 configuration level, the route map will apply to all routes associated with the neighbor.

Although the second route map enables dampening, the first route map is still required. The second route map enables dampening for the neighbors to which the route map is applied. However, unless dampening is already enabled globally by the first route map, the second route map has no effect.

The last two commands apply the route maps. The **dampening route-map** command applies the first route map, which enables dampening globally. The **neighbor** command applies the second route map to neighbor 10.10.10.1. Since the second route map does not contain match clauses for specific routes, the route map enables dampening for all routes received from the neighbor.

Removing route dampening from a route

You can un-suppress routes by removing route flap dampening from the routes. The device allows you to un-suppress all routes at once or un-suppress individual routes.

To un-suppress all the suppressed routes, enter the following command at the Privileged EXEC level of the CLI.

```
device# clear ip bgp dampening
```

Syntax: `clear ip bgp dampening [ip-addr ip-mask]`

The **ip-addr** parameter specifies a particular network.

The **ip-mask** parameter specifies the network mask.

To un-suppress a specific route, enter a command such as the following.

```
device# clear ip bgp dampening 10.157.22.0 255.255.255.0
```

This command un-suppresses only the routes for network 10.157.22.0/24.

Displaying and clearing route flap dampening statistics

The software provides many options for displaying and clearing route flap statistics.

Displaying route flap dampening statistics

To display route dampening statistics or all the dampened routes, enter the following command at any CLI level.

```
device# show ip bgp flap-statistics
Total number of flapping routes: 414
  Status Code  >:best d:damped h:history *:valid
  Network      From      Flaps Since      Reuse      Path
h> 10.50.206.0/23 10.90.213.77 1      0 :0 :13 0 :0 :0 65001 4355 1 701
h> 10.255.192.0/20 10.90.213.77 1      0 :0 :13 0 :0 :0 65001 4355 1 7018
h> 10.252.165.0/24 10.90.213.77 1      0 :0 :13 0 :0 :0 65001 4355 1 7018
h> 10.50.208.0/23 10.90.213.77 1      0 :0 :13 0 :0 :0 65001 4355 1 701
h> 10.33.0.0/16 10.90.213.77 1      0 :0 :13 0 :0 :0 65001 4355 1 701
*> 10.17.220.0/24 10.90.213.77 1      0 :1 :4 0 :0 :0 65001 4355 701 62
```

Syntax: `show ip bgp flap-statistics [regular-expression regular-expression | address mask [longer-prefixes] | neighbor ip-addr] as-path-filter num`

The **regular-expression** *regular-expression* parameter is a regular expression. Regular expressions are the same ones supported for BGP4 AS-path filters.

The **address mask** parameters specify a particular route. If you also use the optional **longer-prefixes** parameter, all statistics for routes that match the specified route or have a longer prefix than the specified route are displayed. For example, if you specify **10.157.0.0 longer**, all routes with the prefix 10.157. or longer (such as 10.157.22.) are displayed.

The **neighborip-addr** parameter displays route flap dampening statistics only for routes learned from the specified neighbor. You also can display route flap statistics for routes learned from a neighbor by entering the following command: **show ip bgp neighbor flap-statistics**.

The **as-path-filter num** parameter specifies one or more filters. Only the routes that have been dampened and that match the specified filter or filters are displayed.

TABLE 46 show ip bgp flap-statistics output descriptions

This field	Displays
Total number of flapping routes	The total number of routes in the BGP4 route table that have changed state and have been marked as flapping routes.
Status code	Indicates the dampening status of the route, which can be one of the following: <ul style="list-style-type: none"> • > - This is the best route among those in the BGP4 route table to the route destination. • d - This route is currently dampened, and unusable. • h - The route has a history of flapping and is unreachable now. • * - The route has a history of flapping but is currently usable.
Network	The destination network of the route.
From	The neighbor that sent the route to the device.
Flaps	The number of flaps the route has experienced.
Since	The amount of time since the first flap of this route.
Reuse	The amount of time remaining until this route will be un-suppressed and can be used again.
Path	Shows the AS-path information for the route.

You also can display all dampened routes by entering the **show ip bgp dampened-paths** command.

Clearing route flap dampening statistics

Clearing the dampening statistics for a route does not change the dampening status of the route. To clear all the route dampening statistics, enter the following command at any level of the CLI.

```
device# clear ip bgp flap-statistics
```

Syntax: **clear ip bgp flap-statistics** [**regular-expression** *regular-expression* | **address mask** | **neighbor ip-addr**]

The parameters are the same as those for the **show ip bgp flap-statistics** command (except the **longer-prefixes** option is not supported).

NOTE

The **clear ip bgp dampening** command not only clears statistics but also un-suppresses the routes.

Generating traps for BGP4

You can enable and disable SNMP traps for BGP4. BGP4 traps are enabled by default.

To enable BGP4 traps after they have been disabled, enter the following command.

```
device(config)# snmp-server enable traps bgp
```

Syntax: [no] snmp-server enable traps bgp

Use the **no** form of the command to disable BGP4 traps.

Configuring BGP4

Once you activate BGP4, you can configure the BGP4 options. There are two configuration levels: global and address family.

At the *global level*, all BGP4 configurations apply to IPv4 and IPv6. Enter this layer using the **device BGP4** command

Under the global level, you specify an address family. Address families separate IPv4 and IPv6 BGP4 configurations. Go to this level by entering the **address-family** command at the device BGP4 level. The command requires you to specify the IPv4 or IPv6 network protocol.

The **address-family** command also requires you to select a sub-address family, which is the type of routes for the configuration. Specify unicast routes.

TABLE 47 IPv4 BGP4 commands for different configuration levels

Command	Global (IPv4 and IPv6)	IPv4 address family unicast
address-family	x	x
aggregate-address		x
always-compare-med	x	
always-propagate		x
as-path-ignore	x	
bgp-redistribute-internal	x	
client-to-client-reflection	x	x
cluster-id		x
compare-med-empty-aspath	x	
compare-routerid	x	
confederation	x	
dampening		x
default-information-originate		x
default-local-preference	x	
default-metric		x
distance	x	
enforce-first-as	x	
exit-address-family	x	x
fast-external-fallover	x	
graceful-restart		x
install-igp-cost		x
local-as	x	
log-dampening-debug		x
maxas-limit		x
maximum-paths		x
med-missing-as-worst	x	
multipath		x

Entering and exiting the address family configuration level

TABLE 47 IPv4 BGP4 commands for different configuration levels (continued)

Command	Global (IPv4 and IPv6)	IPv4 address family unicast
neighbor	x	x
network		x
next-hop-enable-default		x
next-hop-recursion		x
redistribute		x
rib-route-limit		x
show	x	x
static-network		
table-map		x
timers	x	
update-time		x

Entering and exiting the address family configuration level

The BGP4 address family contains a unicast sub-level.

To go to the IPv4 BGP4 unicast address family configuration level, enter the following command.

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

NOTE

The CLI prompt for the global BGP4 level and the BGP4 address-family IPv4 unicast level is the same.

Syntax: `[no] address-family ipv4 unicast [vrf vrf-name]`

The default is the IPv4 unicast address family level.

The **vrf** option allows you to configure a unicast instance for the VRF specified by the *vrf-name* variable.

To exit an address family configuration level, enter the following command.

```
device(config-bgp) # exit-address-family
device(config-bgp) #
```

Syntax: `[no] exit-address-family`

BGP route reflector

A BGP device selects a preferred BGP4 route for a specific prefix learned from multiple peers by using the BGP best path selection algorithm, and installs the BGP4 route in the Routing Table Manager (RTM). The BGP device marks the preferred BGP4 route as the best route, and advertises the route to other BGP4 neighbors. Generally, the RTM route table size is larger than the number of unique BGP4 routes in the BGP4 route table. All preferred BGP4 routes are installed in RTM and are marked as the best BGP4 routes.

However, in certain configurations it is possible that the total number of preferred BGP4 routes may exceed the RTM route table size limit. Therefore, some preferred BGP4 routes may not be installed in the RTM, and the BGP device is not able to forward traffic correctly.

for those BGP4 routes. Those BGP4 routes are not considered as the best BGP4 routes, and are not advertised to other BGP4 neighbors because traffic miss-forwarding or packet drop can occur.

When a BGP device is configured as only a route reflector server, and is not placed directly in the forwarding path, it is possible to mark all preferred BGP4 routes as the best routes to be advertised to other BGP4 neighbors even if the routes are not installed in the RTM. To support the behavior of a BGP device as a route reflector server in such a scenario, use the **always-propagate** command and the **rib-route-limit** command.

NOTE

The **always-propagate** command and the **rib-route-limit** command are supported.

Configuring BGP route reflector

The **always-propagate** command enables a device to mark a preferred BGP4 route not installed in the RTM as the best route, and advertise the route to other BGP4 neighbors. The same process for outbound route policy continues to apply to all best BGP4 routes. The **rib-route-limit** command limits the number of BGP4 Routing Information Base (RIB) routes that can be installed in the RTM. The RTM must be able to reserve enough entries for Interior Gateway Protocol (IGP) routes because the IGP routes are required by BGP4 to resolve BGP4 next-hop entries. If the RTM is not able to reserve enough entries for IGP routes, BGP4 RIB routes can fill the entire RTM with only BGP4 route entries. The **rib-route-limit** command enables IGP and BGP4 route entries to be installed in the RTM.

NOTE

The **always-propagate** command and the **rib-route-limit** command are configurable in any order under the BGP4 address family configuration level.

Perform the following steps to advertise a preferred BGP4 route not installed in the RTM.

1. Configure a BGP4 unicast route. Enter a command such as the following.

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

Syntax: **address-family ipv4 unicast** [**vrf vrf-name**] | **ipv6 unicast**

NOTE

To configure a BGP4 unicast route for a specified VRF instance, use the **vrf vrf-name** parameter. The **vrf vrf-name** parameter allows you to create a VPN routing or forwarding instance specified by the **vrf-name** variable. The **vrf-name** variable specifies the name of the VRF instance you want to create.

2. Enter the **always-propagate** command to enable a preferred BGP4 route (not installed in the RTM) to be advertised to other BGP4 neighbors.

```
device(config-bgp)# always-propagate
```

Syntax: **always-propagate**

- Enter the **rib-route-limit** command to set the maximum number of BGP4 rib routes that can be installed in the RTM.

```
device(config-bgp)# rib-route-limit 500
```

Syntax: **rib-route-limit** *decimal*

The *decimal* variable specifies the maximum number of BGP4 rib routes that can be installed in the RTM. The user may enter any number for the *decimal* variable for the **rib-route-limit** command. By default, there is no limit. If the **rib-route-limit** command is set to 0, no BGP4 routes are installed in the RTM. If a BGP4 route is not installed in the RTM because of the configuration set by the **rib-route-limit** command, the **always-propagate** command must be enabled for preferred BGP4 routes to be advertised to the BGP4 neighbors.

If the **rib-route-limit** command is configured to a value that is below the number of BGP4 routes already installed in the RTM, the following warning message is displayed on the console.

```
device(config-bgp)# rib-route-limit 250
The new limit is below the current bgp rib route count. Please use Clear ip bgp routes command to
remove bgp rib routes.
```

You can only use one of the following commands to clear all BGP4 routes in the RTM, and reset the routes for preferred BGP4 routes to be reinstalled in the RTM. Depending on the type of route the **rib-route-limit** command is used for, select from one of the following commands:

- clear ip bgp routes** command. This command is used to clear IPv4 BGP unicast routes.
- clear ipv6 bgp routes** command. This command is used to clear IPv6 BGP unicast routes.

NOTE

It is not guaranteed that the same number of preferred BGP4 routes will be reinstalled in the RTM.

- Perform the following step to:
 - exit the BGP4 unicast family configuration.

```
device(config-bgp-ipv4u)# exit-address-family
```

Syntax: **exit-address-family**

When you enter the **exit-address-family** command at the address family configuration level, you return to the BGP4 unicast address family configuration level (the default BGP4 level).

Displaying configuration for BGP route reflector

To display the configuration for preferred BGP4 routes not installed in the RTM, use the **show ip bgp route** command as shown in the following example.

```
device(config-bgp)# show ip bgp route
Total number of BGP Routes: 333422
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
...5        10.12.0.0/24   10.100.100.4      100         0         E
          AS_PATH:48 1994 65148 21948 6461 1239 4837 4808 17431 18245...
```

Syntax: **show ip bgp route**

In the previous output, BGP4 receives 333,422 routes and the **rib-route-limit** command is configured to 300,000 routes. The **always-propagate** command has not been enabled. However, because the **rib-route-limit** command is configured to allow for 300,000 routes in the RTM, BGP4 installs only 300,000 routes of the 333,422 routes received in the RTM. When the **always-propagate** command is enabled, a preferred BGP4 route not installed in the RTM is now considered as the best BGP4 route to be advertised to other peers. The route is identified by the letter "b" (for NOT-INSTALLED-BEST) in the Status field. However, when the **always-propagate** command is

not enabled, the status field displays only the default letter "E", as displayed for BGP4 route 10.12.0.0/24. The letter "B" or "b" is missing from the Status field.

NOTE

The description of the status "b: NOT-INSTALLED-BEST" has changed. The status description for "b: NOT-INSTALLED-BEST" is now: The routes received from the neighbor are the best BGP4 routes to their destinations, but were nonetheless not installed in the IP route table due to the **rib-route-limit** option (or RTM route table size limit), and the **always-propagate** option to allow the propagating of those best BGP routes.

NOTE

Traffic loss on a BGP4 route occurs when a device is advertising preferred BGP4 routes not installed in the RTM as part of the forwarding path.

Because the BGP4 route 10.12.0.0/24 is not considered as the best BGP4 route, the route is not advertised to other BGP4 neighbors.

```
device(config-bgp)# show ip bgp route 10.12.0.0/24
Number of BGP Routes matching display condition : 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  10.12.0.0/24  10.100.100.4      100      0      E
   AS_PATH: 48 1994 65148 21948 6461 1239 4837 4808 17431 18245
   Last update to IP routing table: 0h16m2s
   No path is selected as BEST route
```

Syntax: `show ip bgp route ip-address/prefix`

After enabling the **always-propagate** command, the BGP4 route is now considered the best BGP4 route, even though the route is not installed in the RTM. Because the **rib-route-limit** command was configured to allow for only 300,000 routes in the RTM some preferred BGP4 routes are not installed in the RTM, and are not advertised to other BGP4 neighbors. By enabling the **always-propagate** command, the device is now able to advertise those preferred BGP4 routes to other BGP4 neighbors. In the following example, the Status field displays "bE" indicating that the route is now considered the best BGP4 route for forwarding and will be advertised to other BGP4 neighbors.

```
device(config-bgp)# show ip bgp route 10.12.0.0/24
Number of BGP Routes matching display condition : 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  10.12.0.0/24  10.100.100.4      100      0      bE
   AS_PATH: 48 1994 65148 21948 6461 1239 4837 4808 17431 18245
   Last update to IP routing table: 0h12m53s
   Route is to be sent to 1 peers:
   10.0.0.14(6)
```

For an explanation of the fields displayed in the output of the **show ip bgp route** command, refer to [Displaying information for a specific route](#) on page 400.

Specifying a maximum AS path length

You can use the **maxas-limit in** command to configure a device running BGP4 to discard routes that exceed a specified AS path limit. This limit can be configured globally, for peer groups, and for BGP neighbors.

Specifying a maximum AS path length

When you configure **maxas-limit in**, the behavior of the device changes to first check the length of the AS paths in the UPDATE messages and then to apply the inbound policy. If the AS path exceeds the configured length, then the device performs the following actions:

- Does not store the route in the RIB and does not forward the NLRIs and attributes contained in the UPDATE message for that route
- Logs an error
- Processes the withdrawn NLRIs in the same update message

If a route from a peer exceeds the configured Maximum AS path limit, the device also removes the same route from that peer, if it exists, from its own RIB.

After a maximum AS path length is configured, the maximum AS path limit applies to all new inbound routes. To update previously stored routes, you must perform an inbound soft reset for all of the address families activated for that particular BGP neighbor session.

NOTE

If the neighbor soft-reconfiguration feature is enabled, you must perform a hard reset on the device to impose the maximum length limit.

NOTE

maxas-limit in is checked against the received AS_PATH and AS4_PATH attributes.

BGP devices check for and, if configured, apply **maxas-limit in** in the following order:

1. Neighbor value
2. Peer group value
3. Global value

In a case where a neighbor has no maximum AS limit, a peer group has a value of 3 configured, and the system has a value of 9 configured, all of the devices in the peer group will only use the peer group value; the global value will never be used.

Setting a global maximum AS path limit

The syntax for the global maximum AS path limit command is:

Syntax: **[no] maxas-limit in** *num*

The **maxas-limit** keyword specifies the limit on the AS numbers in the as-path attribute. The **in** keyword allows the as-path attribute from any neighbor imposing a limit on AS numbers received. The default maximum length for the global system is 300. The range is 0 - 300. The **no** keyword removes the configuration at the global level.

NOTE

The device applies the BGP4 maximum AS path limit on a per virtual device basis.

To configure the global Maximum AS path limit to 15, enter the following command:

```
device(config-bgp)# maxas-limit in 15
```

Setting a maximum AS path limit for a peer group or neighbor

To set maximum AS path limit for a peer group or a neighbor, the syntax is:

Syntax: **neighbor { ip-addr | peer-group-name } maxas-limit in [num | disable]**

By default, neighbors or peer groups have no configured maximum values. The range is 0 - 300. The **disable** keyword is used to stop a neighbor from inheriting the configuration from the peer-group or global and to use system default value.

To configure a peer group named "PeerGroup1" and set a maximum AS path value of 7, enter the following commands:

```
device(config-bgp)# neighbor PeerGroup1 peer-group
device(config-bgp)# neighbor PeerGroup1 maxas-limit in 7
```

BGP4 max-as error messages

This section lists error log messages that you might see when the device receives routes that exceed the configured AS segment limit or the internal memory limit. The log messages can contain a maximum of 30 ASNs. If a message contains more than 30 ASNs, the message is truncated and an ellipsis appears.

Maximum AS path limit error

```
SYSLOG: <11>Jan 1 00:00:00 mul, BGP: From Peer 192.168.1.2 received Long AS_PATH H= AS_CONFED_SET(4) 1 2 3
AS_CONFED_SEQUENCE(3) 4 AS_SET(1) 5 6 7 AS_SEQ(2) 8 9 attribute length (9) More than configured MAXAS-LIMIT
7
```

Memory limit error

```
SYSLOG: <11>Jan 1 00:00:00 mul, BGP: From Peer 192.168.1.2 received Long AS_PATH H= AS_CONFED_SET(4) 1 2 3
AS_CONFED_SEQUENCE(3) 4 AS_SET(1) 5 6 7 AS_SEQ(2) 8 9 attribute length (9) Exceeded internal memory limit
```

NOTE

The device generates a log message one time every two minutes. Because of this rate limit, it is possible that some errors might not appear in the log. In this case, you can use the **debug ip bgp events** command to view errors pertaining to the **maxas-limit** value and the actual AS path attributes received.

Originating the default route

By default, the 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.

NOTE

The device checks for the existence of an IGP route for 0.0.0.0/0 in the IP route table before creating a local BGP4 route for 0.0.0.0/0.

To configure the device to originate and advertise a default BGP4 route, enter this command.

```
device(config-bgp)# default-information-originate
```

Syntax: [no] default-information-originate

Changing the default metric used for route cost

By default, BGP4 uses the BGP MED value as the route cost when adding the route to the RTM. However, you can configure BGP4 to use the IGP cost instead.

NOTE

It is recommended that you change the default to IGP cost only in mixed-vendor environments, and that you change it on all Brocade devices in the environment.

To change the route cost default from BGP MED to IGP cost, enter a command such as the following:

```
device(config-bgp)# install-igp-cost
```

Syntax: **[no] install-igp-cost**

Use the **no** form of the command to revert to the default of BGP MED.

Configuring a static BGP4 network

This feature 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 RTM (Routing Table Manager). Consequently, when there is a route that can be resolved, it will be installed into the RTM.

To configure a static BGP4 network, enter commands such as the following.

```
device(config)# router bgp
device(config-bgp)# static-network 10.157.22.26/16
```

Syntax: **[no] static-network** *ipAddressPrefix/mask*

The *ipAddress* and *mask* variables are the IPv4 address prefix and mask of the static BGP4 network you are creating.

Using the **no** option uninstalls a route (that was previously installed) from BGP4 RIB-IN and removes the corresponding drop route from the RTM. If there is a new best route, it is advertised to peers if necessary. Otherwise, a withdraw message is sent.

NOTE

The BGP4 network route and the BGP4 static network route are mutually exclusive. They cannot be configured with the same prefix and mask.

When you configure a route using the **static-network** command, BGP4 automatically generates a local route in BGP4 RIB-IN, and installs a NULL0 route in the RTM if there is no other valid route with the same prefix/mask learned from any peer. Otherwise, the learned BGP4 route will be installed in the RTM. In either situation, the new locally generated route will be the best route in RIB-IN and will be advertised to peers if it passes the per-peer outbound policies.

Setting an administrative distance for a static BGP4 network

When a static BGP4 network route is configured, its type is **local BGP4 route** and has a default administrative distance value of 200. To change the administrative distance value, change the value of all local BGP4 routes using the **distance** command at the router bgp level of the CLI, and set a new value for local routes. You can also assign a specific administrative distance value for each static network using the **distance** option as shown.

```
device(config)# router bgp
device(config-bgp)# static-network 10.157.22.26/16 distance 100
```

Syntax: **[no] static-network** *ipAddressPrefix/mask distance distance-value*

The *ipAddress* and *mask* variables are the IPv4 address prefix and mask of the static BGP4 network for which you are setting an administrative distance.

The *distance-value* sets the administrative distance of the static BGP4 network route. The range for this value is 1 - 255.

Limiting advertisement of a static BGP4 network to selected neighbors

You can control the advertisement of a static BGP4 network to BGP4 neighbors that are configured as Service Edge Devices. When this feature is configured for a BGP4 neighbor, static BGP4 network routes that are installed in the routing table as DROP routes are not advertised to that neighbor. When this feature is configured, the route is only advertised to identified Service Edge devices if it is installed as a forward route, such as the routes described in these steps.

1. There is a learned route from a customer BGP4 peering.
2. There is a valid learned route from another Services Edge device as a result of a customer route present on that device.

To configure a BGP4 neighbor to limit the advertisement of Static BGP4 Network routes, enter the **static-network-edge** command as shown.

```
device(config)# router bgp
device(config-bgp)# neighbor 10.2.3.4 static-network-edge
```

Syntax: [no] neighbor *ip-address* | *peer-group-name* static-network-edge

The *ip-addr* and *peer-group-name* variables indicate whether you are configuring an individual neighbor or a peer group. If you specify a neighbor IP address, you are configuring that individual neighbor. If you specify a peer group name, you are configuring a peer group.

Dynamic route filter update

Routing protocols use various route filters to control the distribution of routes. Route filters are used to filter routes received from and advertised to other devices. Protocols also use route-map policies to control route redistribution from other routing protocols. In addition, route filter policies are used to select routes to be installed in the routing tables, and used by forwarding engine to forward traffic.

There are currently 5 different types of route filters defined for use in a device:

- Access List (ACL)
- Prefix-List
- BGP4 as-path Access-list
- BGP4 community-list
- Route-map

Not every protocol uses all of these route filters. A protocol will usually use two or three filter types.

TABLE 48 Route filters used by each protocol

Protocol	Route map	Prefix list	Community- list	As-path access- list	ACL
BGP4	X	X	BGP4 does not use Community- List filters directly. It does use them indirectly through route-map filters that contain Community- List filters.	X	X
OSPF	X	X	X	X	X
RIP	X	X	X	X	
RIPng		X			
OSPFv3	X	X	X	X	
MSDP	X				
MCast					X

When a route filter is changed (created, modified or deleted) by a user, the filter change notification will be sent to all relevant protocols, so that protocols can take appropriate actions. For example if BGP4 is using a route-map (say MapX) to control the routes advertised to a particular peer, the change of route-map (MapX) will cause BGP4 to re-evaluate the advertised routes, and make the appropriate advertisements or withdrawals according to the new route-map policy.

A route filter change action can happen in three ways.

1. A new filter is defined (created).

This filter name may be already referenced by an application. The application needs to be notified of the addition of the new filter, and will bind to and use the new filter. In general, if a filter name is referenced by an application, but is not actually defined, the application assumes the default **deny** action for the filter.

2. An existing filter is undefined (removed).

If the deleted filter is already used and referenced by an application, the application will unbind itself from the deleted filter.

3. An existing filter is modified (updated).

If the filter is already used and referenced by an application, the application will be notified.

Protocols are automatically notified when a route filter is created, deleted or modified. In addition, when a protocol is notified of a filter change, appropriate steps are taken to apply the new or updated filter to existing routes.

Filter update delay and BGP

The **filter-changes-update-delay** command applies (remove only) to changes of filters that are already used or referenced by applications. If the content of a filter is changed, the new filter action takes effect after **filter-changes-update-delay** for existing routes. The notification delay also applies to situations where the usage or reference of a filter is changed in BGP.

For example, the following BGP neighbor command sets or changes the route-map filter on a neighbor:

```
device(config-bgp)# neighbor x.x.x.x route-map map_abc out
```

In this case, the device applies the route-map "map_abc" to the peer, and updates the neighbor out-bound routes after a delay.

If the *delay-time* is 0, BGP does not start peer out-bound policy updates immediately.

Use the **clear filter-change-update** or **clear ip bgp neighbor soft-out** commands to trigger BGP policy updates.

Similarly, the **filter-changes-update-delay** command also applies to the neighbor in-bound policy change.

The general guideline is to define a policy *first*, then apply it to a BGP peer.

BGP4 policy processing order

The order of application of policies when processing inbound and outbound route advertisements on the device is:

1. Ip prefix-list
2. Outbound Ip prefix-list ORF, if negotiated
3. Filter-list (using As-path access-list)
4. Distribute list (using IP ACL - IPv4 unicast only)
5. Route-map

Generalized TTL Security Mechanism support

The device supports the Generalized TTL Security Mechanism (GTSM) as defined in RFC 3682. GTSM protects the device from attacks of invalid BGP4 control traffic that is sent to overload the CPU or hijack the BGP4 session. GTSM protection applies to EBGP neighbors only.

When GTSM protection is enabled, BGP4 control packets sent by the device to a neighbor have a Time To Live (TTL) value of 255. In addition, the device expects the BGP4 control packets received from the neighbor to have a TTL value of either 254 or 255. For multihop peers (where the **ebgp-multihop** option is configured for the neighbor), the device expects the TTL for BGP4 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 BGP4 control packets received from the neighbor do not have the anticipated value, the device drops them.

For more information on GTSM protection, refer to RFC 3682.

To enable GTSM protection for neighbor 192.168.9.210 (for example), enter the following command.

```
device(config-bgp-router)# neighbor 192.168.9.210 ebgp-btsh
```

Syntax: **[no]** **neighbor** *ip-addr* | *peer-group-name* **ebgp-btsh**

NOTE

For GTSM protection to work properly, it must be enabled on both the device and the neighbor.

Displaying BGP4 information

You can display the following configuration information and statistics for BGP4 protocol:

- Summary BGP4 configuration information for the device
- Active BGP4 configuration information (the BGP4 information in the running configuration)
- Neighbor information
- Peer-group information
- Information about the paths from which BGP4 selects routes
- Summary BGP4 route information
- Virtual Routing and Forwarding (VRF) instance information
- The device's BGP4 route table
- Route flap dampening statistics
- Active route maps (the route map configuration information in the running configuration)
- BGP4 graceful restart neighbor Information
- AS4 support and asdot notation

Displaying summary BGP4 information

You can display the local AS number, the maximum number of routes and neighbors supported, and some BGP4 statistics. You can also display BGP4 memory usage for:

- BGP4 routes installed
- Routes advertising to all neighbors (aggregated into peer groups)
- Attribute entries installed

The **show ip bgp summary** command output has the following limitations:

- If a BGP4 peer is not configured for an address-family, the peer information is not displayed.
- If a BGP4 peer is configured for an address-family but not negotiated for an address-family after the BGP4 peer is in the established state, the **show ip bgp summary** command output shows (**NoNeg**) at the end of the line for this peer.
- If a BGP4 peer is configured and negotiated for that address-family, its display is the same as in previous releases.

To view summary BGP4 information for the device, enter the following command at any CLI prompt

```
device# show ip bgp summary
BGP4 Summary
Router ID: 10.10.1.14 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: 67, UP: 67
Number of Routes Installed: 258088, Uses 22195568 bytes
Number of Routes Advertising to All Neighbors:17,035844(3,099146 entries),
Uses 192,147052 bytes
Number of Attribute Entries Installed: 612223, Uses 55100070 bytes
Neighbor Address  AS#   State  Time    Rt:Accepted  Filtered  Sent  ToSend
10.0.100.2        100  ESTABp 0h28m24s  0           0        258087 0
10.0.101.2        100  ESTAB  0h28m24s  0           0        258087 0
10.2.3.4          200  ADMDN  0h44m56s  0           0         0      2
```

Syntax: show ip bgp summary

TABLE 49 show ip bgp summary output descriptions

This field	Displays
Router ID	The device ID.
Local AS Number	The BGP4 AS number for the device.
Confederation Identifier	The AS number of the confederation in which the device resides.
Confederation Peers	The numbers of the local autonomous systems contained in the confederation. This list matches the confederation peer list you configure on the device.
Maximum Number of Paths Supported for Load Sharing	The maximum number of route paths across which the device can balance traffic to the same destination. The feature is enabled by default but the default number of paths is 1. You can increase the number from 2 through 8 paths.
Number of Neighbors Configured	The number of BGP4 neighbors configured on this device, and currently in established state.
Number of Routes Installed	The number of BGP4 routes in the device BGP4 route table and the route or path memory usage.
Number of Routes Advertising to All Neighbors	The total of the RtSent and RtToSent columns for all neighbors, the total number of unique ribout group entries, and the amount of memory used by these groups.
Number of Attribute Entries Installed	The number of BGP4 route-attribute entries in the device route-attributes table and the amount of memory used by these entries.
Neighbor Address	The IP addresses of the BGP4 neighbors for this device.
AS#	The AS number.
State	The state of device sessions with each neighbor. The states are from this perspective of the device, not the neighbor. State values are based on the BGP4 state machine values described in RFC 1771 and can be one of the following for each device: <ul style="list-style-type: none"> • IDLE - The BGP4 process is waiting to be started. Usually, enabling BGP4 or establishing a neighbor session starts the

TABLE 49 show ip bgp summary output descriptions (continued)

This field	Displays
	<p>BGP4 process. A minus sign (-) indicates that the session has gone down and the software is clearing or removing routes.</p> <ul style="list-style-type: none"> • ADMND - The neighbor has been administratively shut down. • CONNECT - BGP4 is waiting for the connection process for the TCP neighbor session to be completed. • ACTIVE - BGP4 is waiting for a TCP connection from the neighbor. Note : If the state frequently changes between CONNECT and ACTIVE, there may be a problem with the TCP connection. • OPEN SENT - BGP4 is waiting for an Open message from the neighbor. • OPEN CONFIRM - BGP4 has received an Open message from the neighbor and is now waiting for either a KEEPALIVE or NOTIFICATION message. If the device receives a KEEPALIVE message from the neighbor, the state changes to Established. If the message is a NOTIFICATION, the state changes to Idle. • ESTABLISHED - BGP4 is ready to exchange UPDATE packets with the neighbor. <p>Operational States:</p> <p>Additional information regarding the operational states of the BGP4 states described above may be added as described in the following:</p> <ul style="list-style-type: none"> • (+) - is displayed if there is more BGP4 data in the TCP receiver queue. Note : If you display information for the neighbor using the show ip bgp neighbor ip-addr command, the TCP receiver queue value will be greater than 0. • (-) - indicates that the session has gone down and the software is clearing or removing routes. • (*) - indicates that the inbound or outbound policy is being updated for the peer. • (s) - indicates that the peer has negotiated restart, and the session is in a stale state. • (r) - indicates that the peer is restarting the BGP4 connection, through restart. • (^) - on the standby MP indicates that the peer is in the ESTABLISHED state and has received restart capability (in the primary MP). • (<) - indicates that the device is waiting to receive the "End of RIB" message the peer. • (p) - indicates that the neighbor ribout group membership change is pending or in progress • () - indicates that the device is waiting to receive the "End of RIB" message the peer
Time	The time that has passed since the state last changed.
Accepted	The number of routes received from the neighbor that this device installed in the BGP4 route table. Usually, this number is lower than the RoutesRcvd number. The difference indicates that this device filtered out some of the routes received in the UPDATE messages.
Filtered	<p>The routes or prefixes that have been filtered out:</p> <ul style="list-style-type: none"> • If soft reconfiguration is enabled, this field shows how many routes were filtered out (not placed in the BGP4 route table) but retained in memory.

TABLE 49 show ip bgp summary output descriptions (continued)

This field	Displays
	<ul style="list-style-type: none"> If soft reconfiguration is not enabled, this field shows the number of BGP4 routes that have been filtered out.
Sent	The number of BGP4 routes the device has sent to the neighbor.
ToSend	The number of routes the device has queued to advertise and withdraw to a neighbor.

Displaying the active BGP4 configuration

To view the active BGP4 configuration information contained in the running configuration without displaying the entire running configuration, enter the following command at any level of the CLI.

```
device# show ip bgp config
router bgp
  local-as 200
  neighbor 10.102.1.1 remote-as 200
  neighbor 10.102.1.1 ebgp-multihop
  neighbor 10.102.1.1 update-source loopback 1
  neighbor 192.168.2.1 remote-as 100
  neighbor 10.200.2.2 remote-as 400
  neighbor 2001:db8::1:1 remote-as 200
  neighbor 2001:db8::1:2 remote-as 400
  neighbor 2001:db8::1 remote-as 300

  address-family ipv4 unicast
    no neighbor 2001:db8::1:1 activate
    no neighbor 2001:db8::1:2 activate
    no neighbor 2001:db8::1 activate
  exit-address-family

  address-family ipv6 unicast
    redistribute static
    neighbor 2001:db8::1:1 activate
    neighbor 2001:db8::1:2 activate
    neighbor 2001:db8::1 activate
  exit-address-family
end of BGP configuration
```

Syntax: show ip bgp config

Displaying summary neighbor information

The **show ip bgp neighbor** command output has the following limitations.

1. If BGP4 peer is not configured for an address-family, the peer information will NOT be displayed.

2. If BGP4 peer is configured for an address-family, it will display the same as in previous releases.

To display summary neighbor information, enter a command such as the following at any level of the CLI.

```
device# show ip bgp neighbor 192.168.4.211 routes-summary
1  IP Address: 192.168.4.211
Routes Accepted/Installed:1,  Filtered/Kept:11,  Filtered:11
  Routes Selected as BEST Routes:1
    BEST Routes not Installed in IP Forwarding Table:0
  Unreachable Routes (no IGP Route for NEXTHOP):0
  History Routes:0

NLRIs Received in Update Message:24,  Withdraws:0 (0),  Replacements:1
  NLRIs Discarded due to
    Maximum Prefix Limit:0, AS Loop:0
    Invalid Nexthop:0, Invalid Nexthop Address:0.0.0.0
    Duplicated Originator_ID:0, Cluster_ID:0

Routes Advertised:0,  To be Sent:0,  To be Withdrawn:0
NLRIs Sent in Update Message:0,  Withdraws:0,  Replacements:0

Peer Out of Memory Count for:
  Receiving Update Messages:0, Accepting Routes(NLRI):0
  Attributes:0, Outbound Routes(RIB-out):0
```

Syntax: show ip bgp neighbors [*ip-addr*] [route-summary]

TABLE 50 show ip bgp neighbors route-summary output descriptions

This field	Displays
IP Address	The IP address of the neighbor.
Routes Received	How many routes the device has received from the neighbor during the current BGP4 session: <ul style="list-style-type: none"> Accepted or Installed - Number of received routes the device accepted and installed in the BGP4 route table. Filtered or Kept - Number of routes that were filtered out, but were retained in memory for use by the soft reconfiguration feature. Filtered - Number of received routes filtered out.
Routes Selected as BEST Routes	The number of routes that the device selected as the best routes to their destinations.
BEST Routes not Installed in IP Forwarding Table	The number of routes received from the neighbor that are the best BGP4 routes to their destinations, but were not installed in the IP route table because the device received better routes from other sources (such as OSPF, RIP, or static IP routes).
Unreachable Routes	The number of routes received from the neighbor that are unreachable because the device does not have a valid RIP, OSPF, or static route to the next-hop.
History Routes	The number of routes that are down but are being retained for route flap dampening purposes.
NLRIs Received in Update Message	The number of routes received in Network Layer Reachability (NLRI) format in UPDATE messages: <ul style="list-style-type: none"> Withdraws - Number of withdrawn routes the device has received. Replacements - Number of replacement routes the device has received.
NLRIs Discarded due to	Indicates the number of times the device discarded an NLRI for the neighbor due to the following reasons: <ul style="list-style-type: none"> Maximum Prefix Limit - The configured maximum prefix amount had been reached.

TABLE 50 show ip bgp neighbors route-summary output descriptions (continued)

This field	Displays
	<ul style="list-style-type: none"> AS Loop - An AS loop occurred. An AS loop occurs when the BGP4 AS-path attribute contains the local AS number. maxas-limit aspath - The number of route entries discarded because the AS path exceeded the configured maximum length or exceeded the internal memory limits. Invalid Nexthop - The next-hop value was not acceptable. Duplicated Originator_ID - The originator ID was the same as the local device ID. Cluster_ID - The cluster list contained the local cluster ID, or the local device ID if the cluster ID is not configured.
Routes Advertised	<p>The number of routes the device has advertised to this neighbor:</p> <ul style="list-style-type: none"> To be Sent - The number of routes queued to send to this neighbor. To be Withdrawn - The number of NLRI for withdrawing routes the device has queued to send to this neighbor in UPDATE messages.
NLRIs Sent in Update Message	<p>The number of NLRI for new routes the device has sent to this neighbor in UPDATE messages:</p> <ul style="list-style-type: none"> Withdraws - Number of routes the device has sent to the neighbor to withdraw. Replacements - Number of routes the device has sent to the neighbor to replace routes the neighbor already has.
Peer Out of Memory Count for	<p>Statistics for the times the device has run out of BGP4 memory for the neighbor during the current BGP4 session:</p> <ul style="list-style-type: none"> Receiving Update Messages - The number of times UPDATE messages were discarded because there was no memory for attribute entries. Accepting Routes (NLRI) - The number of NLRI discarded because there was no memory for NLRI entries. This count is not included in the Receiving Update Messages count. Attributes - The number of times there was no memory for BGP4 attribute entries. Outbound Routes (RIB-out) - The number of times there was no memory to place a "best" route into the neighbor route information base (Adj-RIB-Out) for routes to be advertised.

Displaying BGP4 neighbor information

You can display configuration information and statistics for BGP4 neighbors of the device.

To view BGP4 neighbor information, including the values for all the configured parameters, enter the following command.

NOTE

The display shows all the configured parameters for the neighbor. Only the parameters that have values different from their defaults are shown.

```
device(config-bgp)# show ip bgp neighbors 10.4.0.2
Total number of BGP neighbors:
1  IP Address: 10.4.0.2, AS: 5 (EBGP), RouterID: 10.0.0.1
   Description: neighbor 10.4.0.2
Local AS: 101
State: ESTABLISHED, Time: 0h1m0s, KeepAliveTime: 0, HoldTime: 0
PeerGroup: pgl
Multihop-EBGP: yes, ttl: 1
```

```

RouteReflectorClient: yes
SendCommunity: yes
NextHopSelf: yes
DefaultOriginate: yes (default sent)
MaximumPrefixLimit: 90000
RemovePrivateAs: : yes
RefreshCapability: Received
Route Filter Policies:
  Distribute-list: (out) 20
  Filter-list: (in) 30
  Prefix-list: (in) pfl
  Route-map: (in) setnp1 (out) setnp2
Messages:      Open      Update      KeepAlive      Notification      Refresh-Req
  Sent       : 1         1         1         0         0
  Received: 1         8         1         0         0
Last Update Time: NLRI      Withdraw      NLRI      Withdraw
                  Tx: 0h0m59s  ---              Rx: 0h0m59s  ---
Last Connection Reset Reason:Unknown
Notification Sent:      Unspecified
Notification Received: Unspecified
TCP Connection state: ESTABLISHED
Local host: 10.4.0.1, Local Port: 179
Remote host: 10.4.0.2, Remote Port: 8053
ISentSeq: 52837276 SendNext: 52837392 TotUnAck: 0
TotSent: 116 ReTrans: 0 UnAckSeq: 52837392
IRcvSeq: 2155052043 RcvNext: 2155052536 SendWnd: 16384
TotalRcv: 493 DupliRcv: 0 RcvWnd: 16384
SendQue: 0 RcvQue: 0 CngstWnd: 1460

```

This example shows how to display information for a specific neighbor, by specifying the neighbor's IP address with the command. Since none of the other display options are used, all of the information is displayed for the neighbor. The number in the far left column indicates the neighbor for which information is displayed. When you list information for multiple neighbors, this number makes the display easier to read.

The TCP statistics at the end of the display show status for the TCP session with the neighbor. Most of the fields show information stored in the Transmission Control Block (TCB) for the TCP session between the device and the neighbor. These fields are described in detail in section 3.2 of RFC 793, "Transmission Control Protocol Functional Specification".

Syntax: `show ip bgp neighbors [ip-addr [advertised-routes [detail [ip-add [/ mask-bits]]]] [attribute-entries [detail]] [flap-statistics] [last-packet-with-error] [received prefix-filter] [received-routes] [routes [best] [detail [best]] [not-installed-best] [unreachable]] [rib-out-routes [ip-addr/mask-bits | ip-addr net-mask | detail]] [routes-summary]]`

The *ip-addr* option lets you narrow the scope of the command to a specific neighbor.

The **advertised-routes** option displays only the routes that the device has advertised to the neighbor during the current BGP4 session.

The **attribute-entries** option shows the attribute-entries associated with routes received from the neighbor.

The **flap-statistics** option shows the route flap statistics for routes received from or sent to the neighbor.

The **last-packet-with-error** option displays the last packet from the neighbor that contained an error. The packet contents are displayed in decoded (human-readable) format.

The **received prefix-filter** option shows the Outbound Route Filters (ORFs) received from the neighbor. This option applies to cooperative route filtering.

The **received-routes** option lists all the route information received in route updates from the neighbor since the soft reconfiguration feature was enabled.

The **routes** option lists the routes received in UPDATE messages from the neighbor. You can specify the following additional options:

- **best** - Displays the routes received from the neighbor that the device selected as the best routes to their destinations.

- **not-installed-best** - Displays the routes received from the neighbor that are the best BGP4 routes to their destinations, but were not installed in the IP route table because the device received better routes from other sources (such as OSPF, RIP, or static IP routes).
- **unreachable** - Displays the routes that are unreachable because the device does not have a valid RIP, OSPF, or static route to the next hop.
- **detail** - Displays detailed information for the specified routes. You can refine your information request by also specifying one of the options (**best** , **not-installed-best** , or **unreachable**).

The **rib-out-routes** option lists the route information base (RIB) for outbound routes. You can display all routes or specify a network address.

The **routes-summary** option displays a summary of the following information:

- Number of routes received from the neighbor
- Number of routes accepted by this device from the neighbor
- Number of routes this device filtered out of the UPDATES received from the neighbor and did not accept
- Number of routes advertised to the neighbor
- Number of attribute entries associated with routes received from or advertised to the neighbor.

TABLE 51 show ip bgp neighbor output descriptions

Field	Information displayed
Total Number of BGP4 Neighbors	The number of BGP4 neighbors configured.
IP Address	The IP address of the neighbor.
AS	The AS the neighbor is in.
EBGP or IBGP	Whether the neighbor session is an IBGP session, an EBGP session, or a confederation EBGP session: <ul style="list-style-type: none"> • EBGP - The neighbor is in another AS. • EBGP_Confed - The neighbor is a member of another sub-AS in the same confederation. • IBGP - The neighbor is in the same AS.
RouterID	The neighbor device ID.
Description	The description you gave the neighbor when you configured it on the device.
Local AS	The value (if any) of the Local AS configured.
State	The state of the session with the neighbor. The states are from the device perspective, not the neighbor perspective. The state values are based on the BGP4 state machine values described in RFC 1771 and can be one of the following for each device: <ul style="list-style-type: none"> • IDLE - The BGP4 process is waiting to be started. Usually, enabling BGP4 or establishing a neighbor session starts the BGP4 process. A minus sign (-) indicates that the session has gone down and the software is clearing or removing routes. • ADMND - The neighbor has been administratively shut down. Refer to Administratively shutting down a session with a BGP4 neighbor on page 321. A minus sign (-) indicates that the session has gone down and the software is clearing or removing routes. • CONNECT - BGP4 is waiting for the connection process for the TCP neighbor session to be completed. • ACTIVE - BGP4 is waiting for a TCP connection from the neighbor.

TABLE 51 show ip bgp neighbor output descriptions (continued)

Field	Information displayed
	<p>NOTE If the state frequently changes between CONNECT and ACTIVE, there may be a problem with the TCP connection.</p> <ul style="list-style-type: none"> • OPEN SENT - BGP4 is waiting for an Open message from the neighbor. • OPEN CONFIRM - BGP4 has received an OPEN message from the neighbor and is now waiting for either a KEEPALIVE or NOTIFICATION message. If the device receives a KEEPALIVE message from the neighbor, the state changes to Established. If the message is a NOTIFICATION, the state changes to Idle. • ESTABLISHED - BGP4 is ready to exchange UPDATE messages with the neighbor. <p>If there is more BGP4 data in the TCP receiver queue, a plus sign (+) is also displayed.</p> <p>NOTE If you display information for the neighbor using the show ip bgp neighbor ip-addr command, the TCP receiver queue value will be greater than 0.</p>
Time	The amount of time this session has been in the current state.
KeepAliveTime	The keep alive time, which specifies how often this device sends keepalive messages to the neighbor.
HoldTime	The hold time, which specifies how many seconds the device will wait for a keepalive or update message from a BGP4 neighbor before deciding that the neighbor is not operational.
PeerGroup	The name of the peer group the neighbor is in, if applicable.
Multihop-EBGP	Whether this option is enabled for the neighbor.
RouteReflectorClient	Whether this option is enabled for the neighbor.
SendCommunity	Whether this option is enabled for the neighbor.
NextHopSelf	Whether this option is enabled for the neighbor.
DefaultOriginate	Whether this option is enabled for the neighbor.
MaximumPrefixLimit	Maximum number of prefixes the device will accept from this neighbor.
RemovePrivateAs	Whether this option is enabled for the neighbor.
RefreshCapability	Whether this device has received confirmation from the neighbor that the neighbor supports the dynamic refresh capability.
CooperativeFilteringCapability	Whether the neighbor is enabled for cooperative route filtering.
Distribute-list	Lists the distribute list parameters, if configured.
Filter-list	Lists the filter list parameters, if configured.
Prefix-list	Lists the prefix list parameters, if configured.
Route-map	Lists the route map parameters, if configured.
Messages Sent	<p>The number of messages this device has sent to the neighbor. The display shows statistics for the following message types:</p> <ul style="list-style-type: none"> • Open • Update • KeepAlive • Notification • Refresh-Req

TABLE 51 show ip bgp neighbor output descriptions (continued)

Field	Information displayed
Messages Received	The number of messages this device has received from the neighbor. The message types are the same as for the Message Sent field.
Last Update Time	Lists the last time updates were sent and received for the following: <ul style="list-style-type: none"> • NLRI • Withdraws
Last Connection Reset Reason	<p>The reason the previous session with this neighbor ended. The reason can be one of the following:</p> <p>Reasons described in the BGP4 specifications:</p> <ul style="list-style-type: none"> • Message Header Error • Connection Not Synchronized • Bad Message Length • Bad Message Type • OPEN Message Error • Unsupported Version Number • Bad Peer AS Number • Bad BGP4 Identifier • Unsupported Optional Parameter • Authentication Failure • Unacceptable Hold Time • Unsupported Capability • UPDATE Message Error • Malformed Attribute List • Unrecognized Well-known Attribute • Missing Well-known Attribute • Attribute Flags Error • Attribute Length Error • Invalid ORIGIN Attribute • Invalid NEXT_HOP Attribute • Optional Attribute Error • Invalid Network Field • Malformed AS_PATH • Hold Timer Expired • Finite State Machine Error • Rcv Notification
Last Connection Reset Reason (cont.)	<p>Reasons specific to the Brocade implementation:</p> <ul style="list-style-type: none"> • Reset All Peer Sessions • User Reset Peer Session • Port State Down • Peer Removed • Peer Shutdown • Peer AS Number Change • Peer AS Confederation Change • TCP Connection KeepAlive Timeout • TCP Connection Closed by Remote • TCP Data Stream Error Detected

TABLE 51 show ip bgp neighbor output descriptions (continued)

Field	Information displayed
Notification Sent	<p>If the device receives a NOTIFICATION message from the neighbor, the message contains an error code corresponding to one of the following errors. Some errors have subcodes that clarify the reason for the error. Where applicable, the subcode messages are listed underneath the error code messages.</p> <ul style="list-style-type: none"> • Message Header Error: <ul style="list-style-type: none"> – Connection Not Synchronized – Bad Message Length – Bad Message Type – Unspecified • Open Message Error: <ul style="list-style-type: none"> – Unsupported Version – Bad Peer As – Bad BGP4 Identifier – Unsupported Optional Parameter – Authentication Failure – Unacceptable Hold Time – Unspecified • Update Message Error: <ul style="list-style-type: none"> – Malformed Attribute List – Unrecognized Attribute – Missing Attribute – Attribute Flag Error – Attribute Length Error – Invalid Origin Attribute – Invalid NextHop Attribute – Optional Attribute Error – Invalid Network Field – Malformed AS Path – Unspecified • Hold Timer Expired • Finite State Machine Error • Cease • Unspecified
Notification Received	Refer to details for the field Notification Sent.
TCP Connection state	<p>The state of the connection with the neighbor. The connection can have one of the following states:</p> <ul style="list-style-type: none"> • LISTEN - Waiting for a connection request. • SYN-SENT - Waiting for a matching connection request after having sent a connection request. • SYN-RECEIVED - Waiting for a confirming connection request acknowledgment after having both received and sent a connection request. • ESTABLISHED - Data can be sent and received over the connection. This is the normal operational state of the connection. • FIN-WAIT-1 - Waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request previously sent. • FIN-WAIT-2 - Waiting for a connection termination request from the remote TCP. • CLOSE-WAIT - Waiting for a connection termination request from the local user.

TABLE 51 show ip bgp neighbor output descriptions (continued)

Field	Information displayed
	<ul style="list-style-type: none"> CLOSING - Waiting for a connection termination request acknowledgment from the remote TCP. LAST-ACK - Waiting for an acknowledgment of the connection termination request previously sent to the remote TCP (which includes an acknowledgment of its connection termination request). TIME-WAIT - Waiting for enough time to pass to be sure the remote TCP received the acknowledgment of its connection termination request. CLOSED - There is no connection state.
Byte Sent	The number of bytes sent.
Byte Received	The number of bytes received.
Local host	The IP address of the device.
Local port	The TCP port the device is using for the BGP4 TCP session with the neighbor.
Remote host	The IP address of the neighbor.
Remote port	The TCP port the neighbor is using for the BGP4 TCP session with the device.
ISentSeq	The initial send sequence number for the session.
SendNext	The next sequence number to be sent.
TotUnAck	The number of sequence numbers sent by the device that have not been acknowledged by the neighbor.
TotSent	The number of sequence numbers sent to the neighbor.
ReTrans	The number of sequence numbers that the device retransmitted because they were not acknowledged.
UnAckSeq	The current acknowledged sequence number.
IRcvSeq	The initial receive sequence number for the session.
RcvNext	The next sequence number expected from the neighbor.
SendWnd	The size of the send window.
TotalRcv	The number of sequence numbers received from the neighbor.
DupliRcv	The number of duplicate sequence numbers received from the neighbor.
RcvWnd	The size of the receive window.
SendQue	The number of sequence numbers in the send queue.
RcvQue	The number of sequence numbers in the receive queue.
CngstWnd	The number of times the window has changed.

Displaying route information for a neighbor

You can display routes based on the following criteria:

- A summary of the routes for a specific neighbor.
- Routes received from the neighbor that the device selected as the best routes to their destinations.
- Routes received from the neighbor that are the best BGP4 routes to their destinations, but were not installed in the IP route table because the device received better routes from other sources (such as OSPF, RIP, or static IP routes).
- Routes that are unreachable because the device does not have a valid RIP, OSPF, or static route to the next hop.
- Routes for a specific network advertised by the device to the neighbor.

- The Routing Information Base (RIB) for a specific network advertised to the neighbor. You can display the RIB regardless of whether the device has already sent it to the neighbor.

Displaying advertised routes

To display the routes the device has advertised to a specific neighbor for a specific network, enter a command such as the following at any level of the CLI.

```
device# show ip bgp neighbors 192.168.4.211 advertised-routes
      There are 2 routes advertised to neighbor 192.168.4.211
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST 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
      Network      Next Hop      Metric    LocPrf    Weight    Status
1      10.102.0.0/24 192.168.2.102    12                32768     BL
2      10.200.1.0/24 192.168.2.102     0                32768     BL
```

You also can enter a specific route.

```
device# show ip bgp neighbors 192.168.4.211 advertised 10.1.1.0/24
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST I:IBGP L:LOCAL
      Network      Next Hop      Metric    LocPrf    Weight    Status
1      10.200.1.0/24 192.168.2.102     0                32768     BL
```

Syntax: `show ip bgp neighbor ip-addr advertised-routes [ip-addr/prefix]`

For information about the fields in this display, refer to [Displaying summary route information](#) on page 396. The fields in this display also appear in the `show ip bgp` display.

Displaying the best routes

To display the routes received from a specific neighbor that are the "best" routes to their destinations, enter a command such as the following at any level of the CLI.

```
device# show ip bgp neighbors 192.168.4.211 routes best
```

Syntax: `show ip bgp neighbors ip-addr routes best`

For information about the fields in this display, refer to [Displaying information for a specific route](#) on page 400. The fields in this display also appear in the `show ip bgp` display.

Displaying the routes with destinations that are unreachable

To display BGP4 routes with destinations that are unreachable using any of the BGP4 paths in the BGP4 route table, enter a command such as the following at any level of the CLI:

```
device(config-bgp)# show ip bgp neighbor 192.168.4.211 routes unreachable
```

Syntax: `show ip bgp neighbor ip-addr routes unreachable`

For information about the fields in this display, refer to [Displaying summary route information](#) on page 396. The fields in this display also appear in the `show ip bgp` display.

Displaying BGP4 information

Displaying the Adj-RIB-Out for a neighbor

To display the current BGP4 Routing Information Base (Adj-RIB-Out) for a specific neighbor and a specific destination network, enter a command such as the following at any level of the CLI.

```
device# show ip bgp neighbor 192.168.4.211 rib-out-routes 192.168.1.0/24
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST 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      Metric      LocPrf      Weight Status
 1    10.200.1.0/24      0.0.0.0          0          101      32768   BL
```

The Adj-RIB-Out contains the routes that the device either has most recently sent to the neighbor or is about to send to the neighbor.

Syntax: `show ip bgp neighbor ip-addr rib-out-routes [ip-addr/prefix]`

For information about the fields in this display, refer to [Displaying summary route information](#) on page 396. The fields in this display also appear in the `show ip bgp` display.

Displaying peer group information

To display peer-group information, enter a command such as the following at the Privileged EXEC level of the CLI.

```
device# show ip bgp peer-group STR
1  BGP peer-group is STR
   Address family : IPV4 Unicast
   activate
   Address family : IPV4 Multicast
   no activate
   Address family : IPV6 Unicast
   no activate
   Address family : IPV6 Multicast
   no activate
   Address family : VPNV4 Unicast
   no activate
   Address family : L2VPN VPLS
   no activate
Members:
  IP Address: 10.1.1.1, AS: 5
```

Syntax: `show ip bgp peer-group [peer-group-name]`

Only the parameters that have values different from their defaults are listed.

Displaying summary route information

To display summary statistics for all the routes in the device's BGP4 route table, enter a command such as the following at any level of the CLI.

```
device# show ip bgp routes summary
Total number of BGP routes (NLRIs) Installed      : 20
Distinct BGP destination networks                 : 20
Filtered BGP routes for soft reconfig              : 100178
Routes originated by this router                   : 2
Routes selected as BEST routes                     : 19
BEST routes not installed in IP forwarding table   : 1
Unreachable routes (no IGP route for NEXTHOP)     : 1
IBGP routes selected as best routes                : 0
EBGP routes selected as best routes                : 17
```

Syntax: `show ip bgp routes summary`

TABLE 52 show ip bgp routes output descriptions

This field	Displays
Total number of BGP4 routes (NLRIs) Installed	Number of BGP4 routes the device has installed in the BGP4 route table.
Distinct BGP4 destination networks	Number of destination networks the installed routes represent. The BGP4 route table can have multiple routes to the same network.
Filtered BGP4 routes for soft reconfig	Number of route updates received from soft-reconfigured neighbors or peer groups that have been filtered out but retained.
Routes originated by this device	Number of routes in the BGP4 route table that this device originated.
Routes selected as BEST routes	Number of routes in the BGP4 route table that this device has selected as the best routes to the destinations.
BEST routes not installed in IP forwarding table	Number of BGP4 routes that are the best BGP4 routes to their destinations but were not installed in the IP route table because the device received better routes from other sources (such as OSPF, RIP, or static IP routes).
Unreachable routes (no IGP route for NEXTHOP)	Number of routes in the BGP4 route table whose destinations are unreachable because the next-hop is unreachable.
IBGP routes selected as best routes	Number of "best" routes in the BGP4 route table that are IBGP routes.
EBGP routes selected as best routes	Number of "best" routes in the BGP4 route table that are EBGP routes.

Displaying VRF instance information

To display VRF instance information, enter a command such as the following at the Privileged EXEC level of the CLI.

```
device# show ip bgp vrf red
Total number of BGP Routes: 2
Status codes: s suppressed, d damped, h history, * valid, > best, i internal, S stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network        Next Hop      RD          MED      LocPrf Weight Path
*>  10.14.14.0/24   0.0.0.0              0        100      32768  ?
*>  10.11.11.11/32  0.0.0.0              0        100      32768  ?
```

Displaying the BGP4 route table

BGP4 uses filters that you define as well as the algorithm described in [How BGP4 selects a path for a route \(BGP best path selection algorithm\)](#) on page 299 to determine the preferred route to a destination. BGP4 sends only the preferred route to the IP table. To view all the learned BGP4 routes, you can display the BGP4 table.

To view the BGP4 route table, enter the following command.

```
device# show ip bgp routes
Total number of BGP Routes: 97371
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
   Prefix          Next Hop      MED          LocPrf      Weight Status
1    10.3.0.0/8     192.168.4.106              100          0        BE
   AS_PATH: 65001 4355 701 80
2    10.4.0.0/8     192.168.4.106              100          0        BE
   AS_PATH: 65001 4355 1
3    10.60.212.0/22 192.168.4.106              100          0        BE
   AS_PATH: 65001 4355 701 1 189
4    10.6.0.0/8     192.168.4.106              100          0        BE
   AS_PATH: 65001 4355 3356 7170 1455
5    10.8.1.0/24    192.168.4.106 0              100          0        BE
   AS_PATH: 65001
```

Syntax: `show ip bgp routes [[network] ip-addr] num [[age secs]] [as-path-access-list num] [[best]] [cidr-only] [community num | no-export | no-advertise | internet | local-as] [community-access-list num] [community-list num] [detail option] [filter-list num,num,...] [next-hop ip-addr] [no-best] [not-installed-best] [prefix-list string] [regular-expression regular-expression] [route-map map-name] [summary] [unreachable]`

The *ip-addr* option displays routes for a specific network. The **network** keyword is optional. You can enter the network address without entering **network** in front of it.

The *num* option specifies the table entry with which you want the display to start. For example, if you want to list entries beginning with table entry 100, specify 100.

The **age secs** parameter displays only the routes that have been received or updated more recently than the number of seconds you specify.

The **as-path-access-list***num* parameter filters the display using the specified AS-path ACL.

The **best** parameter displays the routes received from the neighbor that the device selected as the best routes to their destinations.

The **cidr-only** option lists only the routes whose network masks do not match their class network length.

The **community** option lets you display routes for a specific community. You can specify **local-as** , **no-export** , **no-advertise** , **internet** , or a private community number. You can specify the community number as either two five-digit integer values of up to 1 through 65535, separated by a colon (for example, 12345:6789) or a single long integer value.

The **community-access-list** *num* parameter filters the display using the specified community ACL.

The **community-list** option lets you display routes that match a specific community filter.

The **detail** option lets you display more details about the routes. You can refine your request by also specifying one of the other display options after the detail keyword.

The **filter-list** option displays routes that match a specific address filter list.

The **next-hop** *ip-addr* option displays the routes for a given next-hop IP address.

The **no-best** option displays the routes for which none of the routes to a given prefix were selected as the best route. The IP route table does not contain a BGP4 route for any of the routes listed by the command.

The **not-installed-best** option displays the routes received from the neighbor that are the best BGP4 routes to their destinations, but were not installed in the IP route table because the device received better routes from other sources (such as OSPF, RIP, or static IP routes).

The **prefix-list** *string* parameter filters the display using the specified IP prefix list.

The **regular-expression** *regular-expression* option filters the display based on a regular expression. Refer to [Using regular expressions](#) on page 348.

The **route-map** *map-name* parameter filters the display by using the specified route map. The software displays only the routes that match the match clauses in the route map. Software disregards the route map's set clauses.

The **summary** option displays summary information for the routes.

The **unreachable** option displays the routes that are unreachable because the device does not have a valid RIP, OSPF, or static route to the next-hop.

Displaying the best BGP4 routes

To display all the BGP4 routes in the device's BGP4 route table that are the best routes to their destinations, enter a command such as the following at any level of the CLI

```
device# show ip bgp routes best
Searching for matching routes, use ^C to quit...
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
      E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
      S:SUPPRESSED F:FILTERED s:STALE
Prefix      Next Hop      MED      LocPrf      Weight Status
1    10.3.0.0/8      192.168.4.106      100      0      BE
      AS_PATH: 65001 4355 701 80
2    10.4.0.0/8      192.168.4.106      100      0      BE
      AS_PATH: 65001 4355 1
3    10.60.212.0/22  192.168.4.106      100      0      BE
      AS_PATH: 65001 4355 701 1 189
4    10.6.0.0/8      192.168.4.106      100      0      BE
      AS_PATH: 65001 4355 3356 7170 1455
5    10.2.0.0/16     192.168.4.106      100      0      BE
      AS_PATH: 65001 4355 701
```

Syntax: show ip bgp routes best

Displaying the best BGP4 routes that are not in the IP route table

When the device has multiple routes to a destination from different sources (such as BGP4, OSPF, RIP, or static routes), the device selects the route with the lowest administrative distance as the best route, and installs that route in the IP route table.

To display the BGP4 routes that are the "best" routes to their destinations but are not installed in the device IP route table, enter a command such as the following at any level of the CLI.

```
device#show ip bgp routes not-installed-best
Searching for matching routes, use ^C to quit...
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      Metric      LocPrf      Weight Status
1    192.168.4.0/24    192.168.4.106      0      100      0      bE
      AS_PATH: 65001
```

Each of the displayed routes is a valid path to its destination, but the device received another path from a different source (such as OSPF, RIP, or a static route) that has a lower administrative distance. The device always selects the path with the lowest administrative distance to install in the IP route table.

Notice that the route status in this example is the new status, "b".

Syntax: show ip bgp routes not-installed-best

NOTE

To display the routes that the device has selected as the best routes and installed in the IP route table, display the IP route table using the **show ip route** command.

Displaying BGP4 routes whose destinations are unreachable

To display BGP4 routes whose destinations are unreachable using any of the BGP4 paths in the BGP4 route table, enter a command such as the following at any level of the CLI.

```
device# show ip bgp routes unreachable
Searching for matching routes, use ^C to quit...
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
```

Displaying BGP4 information

```

S:SUPPRESSED F:FILTERED s:STALE
Prefix      Next Hop      Metric      LocPrf      Weight Status
1  10.8.8.0/24  192.168.5.1    0           101         0
AS_PATH: 65001 4355 1

```

Syntax: show ip bgp routes unreachable

Displaying information for a specific route

To display BGP4 network information by specifying an IP address within the network, enter a command such as the following at any level of the CLI.

```

device# show ip bgp 10.3.4.0
Number of BGP Routes matching display condition : 1
Status codes: s suppressed, d damped, h history, * valid, > best, i internal
Origin codes: i - IGP, e - EGP, ? - incomplete
Network      Next Hop      Metric      LocPrf      Weight Path
*> 10.3.4.0/24  192.168.4.106    100         0        65001 4355 1 1221 ?
    Last update to IP routing table: 0h11m38s, 1 path(s) installed:
        Gateway      Port
        192.168.2.1    1/2/1
    Route is advertised to 1 peers:
    10.20.20.2(65300)

```

Syntax: show ip bgp [route] ip-addr/prefix [longer-prefixes] | ip-addr

If you use the **route** option, the display for the information is different, as shown in the following example.

```

device# show ip bgp route 10.3.4.0
Number of BGP Routes matching display condition : 1
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED E:EBGP D:DAMPED
       H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH S:SUPPRESSED F:FILTERED s:STALE
Prefix      Next Hop      MED      LocPrf      Weight Status
1  10.3.4.0/24  192.168.4.106    100         0        BE
AS_PATH: 65001 4355 1 1221
    Last update to IP routing table: 0h12m1s, 1 path(s) installed:
        Gateway      Port
        192.168.2.1    1/2/1
    Route is advertised to 1 peers:
    10.20.20.2(65300)

```

TABLE 53 show ip bgp route output descriptions

This field	Displays
Number of BGP4 Routes matching display condition	The number of routes that matched the display parameters you entered. This is the number of routes displayed by the command.
Status codes	A list of the characters the display uses to indicate the route's status. The status code appears in the left column of the display, to the left of each route. The status codes are described in the command's output. NOTE This field appears only if you <i>do not</i> enter the route option.
Prefix	The network address and prefix.
Next Hop	The next-hop device for reaching the network.
Metric	The value of the route's MED attribute. If the route does not have a metric, this field is blank.
LocPrf	The degree of preference for this route relative to other routes in the local AS. When the BGP4 algorithm compares routes on the basis of local preferences, the route with the higher local preference is chosen. The preference can have a value from 0 - 4294967295.

TABLE 53 show ip bgp route output descriptions (continued)

This field	Displays
Weight	The value that this device associates with routes from a specific neighbor. For example, if the device receives routes to the same destination from two BGP4 neighbors, the device prefers the route from the neighbor with the larger weight.
Path	The route AS path. NOTE This field appears only if you <i>do not</i> enter the route option.
Origin code	A character that indicates the route origin. The origin code appears to the right of the AS path (Path field). The origin codes are described in the command output. NOTE This field appears only if you <i>do not</i> enter the route option.
Status	The route status, which can be one or more of the following: <ul style="list-style-type: none"> • A - AGGREGATE. The route is an aggregate route for multiple networks. • B - BEST. BGP4 has determined that this is the optimal route to the destination. NOTE If the "b" is lowercase, the software was not able to install the route in the IP route table. <ul style="list-style-type: none"> • b - NOT-INSTALLED-BEST. The routes received from the neighbor are the best BGP4 routes to their destinations, but were not installed in the IP route table because the device received better routes from other sources (such as OSPF, RIP, or static IP routes). • C - CONFED_EBGP. The route was learned from a neighbor in the same confederation and AS, but in a different sub-AS within the confederation. • D - DAMPED. This route has been dampened (by the route dampening feature), and is currently unusable. • H - HISTORY. Route dampening is configured for this route, and the route has a history of flapping and is unreachable now. • I - INTERNAL. The route was learned through BGP4. • L - LOCAL. The route originated on this device. • M - MULTIPATH. BGP4 load sharing is enabled and this route was selected as one of the best ones to the destination. The best route among the multiple paths also is marked with "B". NOTE If the "m" is lowercase, the software was not able to install the route in the IP route table. <ul style="list-style-type: none"> • S - SUPPRESSED. This route was suppressed during aggregation and thus is not advertised to neighbors. NOTE This field appears only if you enter the route option.

Displaying route details

This example shows the information displayed when you use the **detail** option. In this example, the information for one route is shown.

```
device# show ip bgp routes detail 2
Number of BGP Routes matching display condition : 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
1    Prefix: 10.5.5.5/32, Status: BE, Age: 0h2m10s
      NEXT_HOP: 10.0.0.1, Metric: 0, Learned from Peer: 10.0.0.1 (3)
      LOCAL_PREF: 100, MED: none, ORIGIN: igp, Weight: 0
      AS_PATH: 3
      Adj_RIB_out count: 2, Admin distance 20
Last update to IP routing table: 0h2m10s, 1 path(s) installed:
Route is advertised to 2 peers:
10.0.0.3(65002)                10.0.0.5(65002)
```

Syntax: show ip bgp routes detail

TABLE 54 show ip bgp routes detail output descriptions

This field	Displays
Total number of BGP4 Routes	The number of BGP4 routes.
Status codes	A list of the characters that indicate route status. The status code is appears in the left column of the display, to the left of each route. The status codes are described in the command's output.
Prefix	The network prefix and mask length.
Status	<p>The route status, which can be one or more of the following:</p> <ul style="list-style-type: none"> A - AGGREGATE. The route is an aggregate route for multiple networks. B - BEST. BGP4 has determined that this is the optimal route to the destination. <p>NOTE If the "b" is lowercase, the software was not able to install the route in the IP route table.</p> <ul style="list-style-type: none"> b - NOT-INSTALLED-BEST. The routes received from the neighbor are the best BGP4 routes to their destinations, but were not installed in the IP route table because the device received better routes from other sources (such as OSPF, RIP, or static IP routes). C - CONFED_EBGP. The route was learned from a neighbor in the same confederation and AS, but in a different sub-AS within the confederation. D - DAMPED. This route has been dampened (by the route dampening feature), and is currently unusable. H - HISTORY. Route dampening is configured for this route, and the route has a history of flapping and is unreachable now. I - INTERNAL. The route was learned through BGP4. L - LOCAL. The route originated on this device. M - MULTIPATH. BGP4 load sharing is enabled and this route was selected as one of the best ones to the destination. The best route among the multiple paths also is marked with "B".

TABLE 54 show ip bgp routes detail output descriptions (continued)

This field	Displays
	<p>NOTE If the "m" is lowercase, the software was not able to install the route in the IP route table.</p> <ul style="list-style-type: none"> S - SUPPRESSED. This route was suppressed during aggregation and thus is not advertised to neighbors.
Age	The last time an update occurred.
Next_Hop	The next-hop device for reaching the network.
Learned from Peer	The IP address of the neighbor that sent this route.
Local_Pref	The degree of preference for this route relative to other routes in the local AS. When the BGP4 algorithm compares routes on the basis of local preferences, the route with the higher local preference is chosen. The preference can have a value from 0 through 4294967295.
MED	The route metric. If the route does not have a metric, this field is blank.
Origin	<p>The source of the route information. The origin can be one of the following:</p> <ul style="list-style-type: none"> EGP - The routes with these attributes came to BGP4 through EGP. IGP - The routes with these attributes came to BGP4 through IGP. INCOMPLETE - The routes came from an origin other than one of the above. For example, they may have been redistributed from OSPF or RIP. <p>When BGP4 compares multiple routes to select the best route, IGP is preferred over EGP and both are preferred over INCOMPLETE.</p>
Weight	The value this device associates with routes from a specific neighbor. For example, if the device receives routes to the same destination from two BGP4 neighbors, the device prefers the route from the neighbor with the larger weight.
Atomic	<p>Whether network information in this route has been aggregated and this aggregation has resulted in information loss.</p> <p>NOTE Information loss under these circumstances is a normal part of BGP4 and does not indicate an error.</p>
Aggregation ID	The device that originated this aggregation.
Aggregation AS	The AS in which the network information was aggregated. This value applies only to aggregated routes and is otherwise 0.
Originator	The originator of the route in a route reflector environment.
Cluster List	The route-reflector clusters through which this route has passed.
Learned From	The IP address of the neighbor from which the device learned the route.
Admin Distance	The administrative distance of the route.
Adj_RIB_out	The number of neighbors to which the route has been or will be advertised. This is the number of times the route has been selected as the best route and placed in the Adj-RIB-Out (outbound queue) for a BGP4 neighbor.
Communities	The communities the route is in.

Displaying BGP4 route-attribute entries

The route-attribute entries table lists the sets of BGP4 attributes stored in device memory. Each set of attributes is unique and can be associated with one or more routes. In fact, the device typically has fewer route attribute entries than routes.

To display the IP route table, enter the following command.

```
device# show ip bgp attribute-entries
```

Syntax: show ip bgp attribute-entries

This example shows the information displayed by this command. A zero value indicates that the attribute is not set.

```
device# show ip bgp attribute-entries
Total number of BGP Attribute Entries: 7753
 1      Next Hop :192.168.11.1      MED :0      Origin:IGP
      Originator:0.0.0.0      Cluster List:None
      Aggregator:AS Number :0      Router-ID:0.0.0.0      Atomic:FALSE
      Local Pref:100      Communities:Internet
      AS Path : (65002) 65001 4355 2548 3561 5400 6669 5548
 2      Next Hop :192.168.11.1      Metric :0      Origin:IGP
      Originator:0.0.0.0      Cluster List:None
      Aggregator:AS Number :0      Router-ID:0.0.0.0      Atomic:FALSE
      Local Pref:100      Communities:Internet
      AS Path : (65002) 65001 4355 2548
```

TABLE 55 show ip bgp attribute-entries output descriptions

This field	Displays
Total number of BGP4 Attribute Entries	The number of routes contained in this BGP4 route table.
Next Hop	The IP address of the next-hop device for routes that have this set of attributes.
Metric	The cost of the routes that have this set of attributes.
Origin	<p>The source of the route information. The origin can be one of the following:</p> <ul style="list-style-type: none"> EGP - The routes with these attributes came to BGP4 through EGP. IGP - The routes with these attributes came to BGP4 through IGP. INCOMPLETE - The routes came from an origin other than one of the above. For example, they may have been redistributed from OSPF or RIP. <p>When BGP4 compares multiple routes to a destination to select the best route, IGP is preferred over EGP and both are preferred over INCOMPLETE.</p>
Originator	The originator of the route in a route reflector environment.
Cluster List	The route-reflector clusters through which this set of attributes has passed.
Aggregator	<p>Aggregator information:</p> <ul style="list-style-type: none"> AS Number shows the AS in which the network information in the attribute set was aggregated. This value applies only to aggregated routes and is otherwise 0. Router-ID shows the device that originated this aggregator.
Atomic	<p>Whether the network information in this set of attributes has been aggregated <i>and</i> this aggregation has resulted in information loss.</p> <ul style="list-style-type: none"> TRUE - Indicates information loss has occurred FALSE - Indicates no information loss has occurred

TABLE 55 show ip bgp attribute-entries output descriptions (continued)

This field	Displays
	NOTE Information loss under these circumstances is a normal part of BGP4 and does not indicate an error.
Local Pref	The degree of preference for routes that use these attributes relative to other routes in the local AS.
Communities	The communities to which routes with these attributes belong.
AS Path	The autonomous systems through which routes with these attributes have passed. The local AS is shown in parentheses.

Displaying the routes BGP4 has placed in the IP route table

The IP route table indicates the routes it has received from BGP4 by listing "BGP" as the route type.

To display the IP route table, enter the following command.

```
device# show ip route
```

Syntax: show ip route [ip-addr | num | bgp | ospf | rip]

This example shows the information displayed by this command. Notice that most of the routes in this example have type "B", indicating that their source is BGP4.

```
device# show ip route
Total number of IP routes: 50834
B:BGP D:Directly-Connected O:OSPF R:RIP S:Static
Network Address NetMask Gateway Port Cost Type
10.0.0.1 255.0.0.0 192.168.13.2 1/1/1 0 B
10.0.0.2 255.0.0.0 192.168.13.2 1/1/1 0 B
10.0.1.1 255.255.128.0 192.168.13.2 1/1/1 0 B
10.1.0.0 255.255.0.0 0.0.0.0 1/1/1 1 D
10.10.11.0 255.255.255.0 0.0.0.0 1/2/24 1 D
10.2.97.0 255.255.255.0 192.168.13.2 1/1/1 0 B
10.3.63.0 255.255.255.0 192.168.13.2 1/1/1 0 B
10.3.123.0 255.255.255.0 192.168.13.2 1/1/1 0 B
10.5.252.0 255.255.254.0 192.168.13.2 1/1/1 0 B
10.6.42.0 255.255.254.0 192.168.13.2 1/1/1 0 B
remaining 50824 entries not shown...
```

Displaying route flap dampening statistics

To display route dampening statistics or all the dampened routes, enter the following command at any level of the CLI.

```
device# show ip bgp flap-statistics
Total number of flapping routes: 414
Status Code >:best d:damped h:history *:valid
Network From Flaps Since Reuse Path
h> 10.50.206.0/23 10.90.213.77 1 0 :0 :13 0 :0 :0 65001 4355 1 701
h> 10.255.192.0/20 10.90.213.77 1 0 :0 :13 0 :0 :0 65001 4355 1 7018
h> 10.252.165.0/24 10.90.213.77 1 0 :0 :13 0 :0 :0 65001 4355 1 7018
h> 10.50.208.0/23 10.90.213.77 1 0 :0 :13 0 :0 :0 65001 4355 1 701
h> 10.33.0.0/16 10.90.213.77 1 0 :0 :13 0 :0 :0 65001 4355 1 701
*> 10.17.220.0/24 10.90.213.77 1 0 :1 :4 0 :0 :0 65001 4355 701 62
```

Syntax: show ip bgp flap-statistics [regular-expression regular-expression | address mask [longer-prefixes] | neighbor ip-addr | filter-list num ...]

The **regular-expression***regular-expression* parameter is a regular expression. The regular expressions are the same ones supported for BGP4 AS-path filters.

The **address mask** parameters specify a particular route. If you also use the optional **longer-prefixes** parameter, all statistics for routes that match the specified route or have a longer prefix than the specified route are displayed. For example, if you specify **10.157.0.0 longer**, all routes with the prefix 10.157 or that have a longer prefix (such as 10.157.22) are displayed.

The **neighborip-addr** parameter displays route flap dampening statistics only for routes learned from the specified neighbor. You can also display route flap statistics for routes learned from a neighbor by entering the **show ip bgp neighbor flap-statistics** command.

The **filter-listnum** parameter specifies one or more filters. Only routes that have been dampened and that match the specified filters are displayed.

TABLE 56 show ip bgp flap-statistics output descriptions

This field	Displays
Total number of flapping routes	The total number of routes in the BGP4 route table that have changed state and have been marked as flapping routes.
Status code	The dampening status of the route, which can be one of the following: <ul style="list-style-type: none"> • > - This is the best route among those in the BGP4 route table to the route destination. • d - This route is currently dampened, and thus unusable. • h - The route has a history of flapping and is unreachable now. • * - The route has a history of flapping but is currently usable.
Network	The destination network of the route.
From	The neighbor that sent the route to this device.
Flaps	The number of flaps (state changes) the route has experienced.
Since	The amount of time since the first flap of this route.
Reuse	The amount of time remaining until this route will be un-suppressed and thus be usable again.
Path	The AS-path information for the route.

You can display all dampened routes by entering the **show ip bgp dampened-paths** command.

Displaying the active route map configuration

You can view the active route map configuration (contained in the running configuration) without displaying the entire running configuration by entering the following command at any level of the CLI.

```
device# show route-map
route-map permitnet4 permit 10
  match ip address prefix-list plist1
route-map permitnet1 permit 1
  match ip address prefix-list plist2
route-map setcomm permit 1
  set community 1234:2345 no-export
route-map test111 permit 111
  match address-filters 11
  set community 11:12 no-export
route-map permit1122 permit 12
  match ip address 11
route-map permit1122 permit 13
  match ip address std_22
```

This example shows that the running configuration contains six route maps. Notice that the match and set statements within each route map are listed beneath the command for the route map itself. In this simplified example, each route map contains only one match or set statement.

To display the active configuration for a specific route map, enter a command such as the following, which specifies a route map name.

```
device# show route-map setcomm
route-map setcomm permit 1
set community 1234:2345 no-export
```

This example shows the active configuration for a route map named "setcomm".

Syntax: `show route-map [map-name]`

Displaying BGP4 graceful restart neighbor information

To display BGP4 restart information for BGP4 neighbors, enter the **show ip bgp neighbors** command.

```
device# show ip bgp neighbors
Total number of BGP Neighbors: 6
1 IP Address: 10.50.50.10, AS: 20 (EBGP), RouterID: 10.10.10.20, VRF: default
State: ESTABLISHED, Time: 0h0m18s, KeepAliveTime: 60, HoldTime: 180
KeepAliveTimer Expire in 34 seconds, HoldTimer Expire in 163 seconds
Minimum Route Advertisement Interval: 0 seconds
RefreshCapability: Received
GracefulRestartCapability: Received
Restart Time 120 sec, Restart bit 0
afi/safi 1/1, Forwarding bit 0
GracefulRestartCapability: Sent
Restart Time 120 sec, Restart bit 0
afi/safi 1/1, Forwarding bit 1
Messages: Open Update KeepAlive Notification Refresh-Req
```

....

Displaying AS4 details

This section describes the use of the following **show** commands, which produce output that includes information about AS4s.

- **show ip bgp neighbor** shows whether the AS4 capability is enabled.
- **show ip bgp attribute-entries** shows AS4 path values.
- **show ip bgp** shows the route entries with two and AS4 path information.
- **show route-map** shows the presence of any AS4 configuration data.
- **show ip as-path-access-lists** shows the presence of any AS4 configuration data.
- **show ip bgp config** shows the presence of any AS4 configuration data.

Route entries with four-byte path information

The **show ip bgp** command without of any optional parameters display AS4 path information.

```
device# show ip bgp
Total number of BGP Routes: 1
Status codes: s suppressed, d damped, h history, * valid, > best, i internal, S
stale
Origin codes: i - IGP, e - EGP, ? - incomplete
Network      Next Hop      Metric LocPrf Weight Path
*> 10.1.1.0/24 192.168.1.5    1      100     0      90000 100 200 65535
```

Displaying BGP4 information

```
65536 65537 65538 65539 75000
```

Syntax: show ip bgp

Current AS numbers

To display current AS numbers, use the **show ip bgp neighbors** command at any level of the CLI.

```
device# show ip bgp neighbors
neighbors          Details on TCP and BGP neighbor connections
Total number of BGP Neighbors: 1
1  IP Address: 192.168.1.1, AS: 7701000 (IBGP), RouterID: 192.168.1.1, VRF: default-vrf
State: ESTABLISHED, Time: 0h3m33s, KeepAliveTime: 60, HoldTime: 180
KeepAliveTimer Expire in 49 seconds, HoldTimer Expire in 177 seconds
Minimal Route Advertisement Interval: 0 seconds
RefreshCapability: Received
Messages:      Open      Update      KeepAlive      Notification      Refresh-Req
Sent          : 1        0          5              0                0
Received: 1    1        5              0                0
Last Update Time: NLRI      Withdraw      NLRI      Withdraw
Tx: ---      ---      Rx: 0h3m33s  ---
Last Connection Reset Reason:Unknown
Notification Sent:      Unspecified
Notification Received: Unspecified
Neighbor NLRI Negotiation:
Peer Negotiated IPV4 unicast capability
Peer configured for IPV4 unicast Routes
Neighbor AS4 Capability Negotiation:
Peer Negotiated AS4 capability
Peer configured for AS4 capability

As-path attribute count: 1
Outbound Policy Group:
ID: 1, Use Count: 1
TCP Connection state: ESTABLISHED, flags:00000044 (0,0)
Maximum segment size: 1460
TTL check: 0, value: 0, rcvd: 64
Byte Sent: 148, Received: 203
Local host: 192.168.1.2, Local Port: 179
Remote host: 192.168.1.1, Remote Port: 8041
ISentSeq: 1656867 SendNext: 1657016 TotUnAck: 0
TotSent: 149 ReTrans: 19 UnAckSeq: 1657016
IRcvSeq: 1984547 RcvNext: 1984751 SendWnd: 64981
TotalRcv: 204 DupliRcv: 313 RcvWnd: 65000
SendQueue: 0 RcvQueue: 0 CngstWnd: 5840
```

Syntax: show ip bgp neighbors

TABLE 57 show ip bgp neighbors output descriptions

Field	Description
Total number of BGP Neighbors	Shows the total number of BGP neighbors.
IP Address	Shows the IPv4 address of the neighbor.
AS	Shows the Autonomous System (AS) in which the neighbor resides.
EBGP or IBGP	Shows whether the neighbor session is an IBGP session, an EBGP session, or a confederation EBGP session: <ul style="list-style-type: none"> EBGP - The neighbor is in another AS. EBGP_Confed - The neighbor is a member of another sub-AS in the same confederation. IBGP - The neighbor is in the same AS.
RouterID	Shows the device ID of the neighbor.
VRF	Shows the status of the VRF instance.

TABLE 57 show ip bgp neighbors output descriptions (continued)

Field	Description
State	<p>Shows the state of the device session with the neighbor. The states are from the device's perspective of the session, not the neighbor's perspective. The state can be one of the following values:</p> <ul style="list-style-type: none"> • IDLE - The BGP4 process is waiting to be started. Usually, enabling BGP4 or establishing a neighbor session starts the BGP4 process. A minus sign (-) indicates that the session has gone down and the software is clearing or removing routes. • ADMND - The neighbor has been administratively shut down. A minus sign (-) indicates that the session has gone down and the software is clearing or removing routes. • CONNECT - BGP4 is waiting for the connection process for the TCP neighbor session to be completed. • ACTIVE - BGP4 is waiting for a TCP connection from the neighbor. <p>NOTE If the state frequently changes between CONNECT and ACTIVE, there may be a problem with the TCP connection.</p> <ul style="list-style-type: none"> • OPEN SENT - BGP4 is waiting for an Open message from the neighbor. • OPEN CONFIRM - BGP4 has received an Open message from the neighbor and is now waiting for either a KeepAlive or Notification message. If the device receives a KeepAlive message from the neighbor, the state changes to ESTABLISHED. If the message is a Notification, the state changes to IDLE. • ESTABLISHED - BGP4 is ready to exchange Update messages with the neighbor. <p>If there is more BGP data in the TCP receiver queue, a plus sign (+) is also displayed.</p>
Time	Shows the amount of time this session has been in its current state.
KeepAliveTime	Shows the keepalive time, which specifies how often this device sends KeepAlive messages to the neighbor.
HoldTime	Shows the hold time, which specifies how many seconds the device will wait for a KeepAlive or Update message from a BGP4 neighbor before deciding that the neighbor is dead.
KeepAliveTimer Expire	Shows the time when the keepalive timer is set to expire.
HoldTimer Expire	Shows the time when the hold timer is set to expire.
Minimal Route Advertisement Interval	Shows the minimum time elapsed between the route advertisements to the same neighbor.
RefreshCapability	Shows whether the device has received confirmation from the neighbor that the neighbor supports the dynamic refresh capability.
Messages Sent and Received	<p>Shows the number of messages this device has sent to and received from the neighbor. The display shows statistics for the following message types:</p> <ul style="list-style-type: none"> • Open • Update • KeepAlive • Notification • Refresh-Req

TABLE 57 show ip bgp neighbors output descriptions (continued)

Field	Description
Last Update Time	Shows the list of last time updates were sent and received for the following: <ul style="list-style-type: none"> NLRIs Withdraws
Last Connection Reset Reason	Shows the reason for ending the previous session with this neighbor. The reason can be one of the following: <ul style="list-style-type: none"> No abnormal error has occurred. Reasons described in the BGP specifications: <ul style="list-style-type: none"> Message Header Error Connection Not Synchronized Bad Message Length Bad Message Type OPEN Message Error Unsupported Version Number Bad Peer AS Number Bad BGP Identifier Unsupported Optional Parameter Authentication Failure Unacceptable Hold Time Unsupported Capability UPDATE Message Error Malformed Attribute List Unrecognized Well-known Attribute Missing Well-known Attribute Attribute Flags Error Attribute Length Error Invalid ORIGIN Attribute Invalid NEXT_HOP Attribute
Last Connection Reset Reason (continued)	<ul style="list-style-type: none"> Reasons described in the BGP specifications (continued): <ul style="list-style-type: none"> Optional Attribute Error Invalid Network Field Malformed AS_PATH Hold Timer Expired Finite State Machine Error Rcv Notification Reset All Peer Sessions User Reset Peer Session Port State Down Peer Removed Peer Shutdown Peer AS Number Change Peer AS Confederation Change TCP Connection KeepAlive Timeout TCP Connection Closed by Remote TCP Data Stream Error Detected
Notification Sent	Shows an error code corresponding to one of the following errors if the device sends a Notification message from the neighbor. Some errors have subcodes that clarify the reason for the error. The subcode messages are listed underneath the error code messages, wherever applicable. <ul style="list-style-type: none"> Message Header Error <ul style="list-style-type: none"> Connection Not Synchronized Bad Message Length Bad Message Type Unspecified

TABLE 57 show ip bgp neighbors output descriptions (continued)

Field	Description
	<ul style="list-style-type: none"> • Open Message Error <ul style="list-style-type: none"> - Unsupported Version - Bad Peer AS - Bad BGP Identifier - Unsupported Optional Parameter - Authentication Failure - Unacceptable Hold Time - Unspecified • Update Message Error <ul style="list-style-type: none"> - Malformed Attribute List - Unrecognized Attribute - Missing Attribute - Attribute Flag Error - Attribute Length Error - Invalid Origin Attribute - Invalid NextHop Attribute - Optional Attribute Error - Invalid Network Field - Malformed AS Path - Unspecified • Hold Timer Expired • Finite State Machine Error • Cease • Unspecified
Notification Received	Shows an error code corresponding to one of the listed errors in the Notification Sent field if the device receives a Notification message from the neighbor.
Neighbor NLRI Negotiation	Shows the state of the device's NLRI negotiation with the neighbor. The states can be one of the following: <ul style="list-style-type: none"> • Peer negotiated IPV4 unicast capability • Peer negotiated IPV6 unicast capability • Peer configured for IPV4 unicast routes • Peer configured for IPV6 unicast routes
Neighbor AS4 Capability Negotiation	Shows the state of the device's AS4 capability negotiation with the neighbor. The states can be one of the following: <ul style="list-style-type: none"> • Peer negotiated AS4 capability • Peer configured for AS4 capability
As-path attribute count	Shows the count of the AS-path attribute.
Outbound Policy Group	Shows the ID and the count used in the outbound policy group.
TCP Connection state	Shows the state of the connection with the neighbor. The connection can have one of the following states: <ul style="list-style-type: none"> • LISTEN - Waiting for a connection request. • SYN-SENT - Waiting for a matching connection request after having sent a connection request. • SYN-RECEIVED - Waiting for a confirming connection request acknowledgment after having both received and sent a connection request. • ESTABLISHED - Data can be sent and received over the connection. This is the normal operational state of the connection.

TABLE 57 show ip bgp neighbors output descriptions (continued)

Field	Description
	<ul style="list-style-type: none"> • FIN-WAIT-1 - Waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request previously sent. • FIN-WAIT-2 - Waiting for a connection termination request from the remote TCP. • CLOSE-WAIT - Waiting for a connection termination request from the local user. • CLOSING - Waiting for a connection termination request acknowledgment from the remote TCP. • LAST-ACK - Waiting for an acknowledgment of the connection termination request previously sent to the remote TCP (which includes an acknowledgment of its connection termination request). • TIME-WAIT - Waiting for the specific time to ensure that the remote TCP received the acknowledgment of its connection termination request. • CLOSED - There is no connection state.
Maximum segment size	Shows the TCP maximum segment size.
TTL check	Shows the TCP TTL check.
Byte Sent	Shows the number of bytes sent.
Byte Received	Shows the number of bytes received.
Local host	Shows the IPv4 address of the device.
Local port	Shows the TCP port that the device is using for the BGP4 TCP session with the neighbor.
Remote host	Shows the IPv4 address of the neighbor.
Remote port	Shows the TCP port the neighbor is using for the BGP4 TCP session with the device.
ISentSeq	Shows the initial send sequence number for the session.
SendNext	Shows the next sequence number to be sent.
TotUnAck	Shows the count of sequence numbers sent by the device that have not been acknowledged by the neighbor.
TotSent	Shows the count of the sequence numbers sent to the neighbor.
ReTrans	Shows the count of the sequence numbers that the device retransmitted because they were not acknowledged.
UnAckSeq	Shows the current acknowledged sequence number.
IRcvSeq	Shows the initial receive sequence number for the session.
RcvNext	Shows the next sequence number expected from the neighbor.
SendWnd	Shows the size of the send window.
TotalRcv	Shows the count of the sequence numbers received from the neighbor.
DupliRcv	Shows the count of the duplicate sequence numbers received from the neighbor.
RcvWnd	Shows the size of the receive window.
SendQue	Shows the count of the sequence numbers in the send queue.
RcvQue	Shows the count of the sequence numbers in the receive queue.
CngstWnd	Shows the number of times the window has changed.

Attribute entries

Use the **show ip bgp attribute-entries** command to see AS4 path values, as the following example illustrates.

```
device# show ip bgp attribute-entries
Total number of BGP Attribute Entries: 18 (0)
1      Next Hop :192.168.1.6      MED :1      Origin:INCOMP
      Originator:0.0.0.0      Cluster List:None
      Aggregator:AS Number :0      Router-ID:0.0.0.0      Atomic:None
      Local Pref:100      Communities:Internet
AS Path :90000 80000 (length 11)
)
      Address: 0x10e4e0c4 Hash:489 (0x03028536), PeerIdx 0
      Links: 0x00000000, 0x00000000, nlri: 0x10f4804a
      Reference Counts: 1:0:1, Magic: 51
2      Next Hop :192.168.1.5      Metric :1      Origin:INCOMP
      Originator:0.0.0.0      Cluster List:None
      Aggregator:AS Number :0      Router-ID:0.0.0.0      Atomic:None
      Local Pref:100      Communities:Internet
AS Path :90000 75000 (length 11)
      Address: 0x10e4e062 Hash:545 (0x0301e8f6), PeerIdx 0
      Links: 0x00000000, 0x00000000, nlri: 0x10f47ff0
      Reference Counts: 1:0:1, Magic: 49
```

Syntax: show ip bgp attribute-entries

Running configuration

AS4s appear in the display of a running configuration, as shown.

```
device# show ip bgp config
Current BGP configuration:
router bgp
  local-as 7701000
  confederation identifier 120000
  confederation peers 80000
  neighbor 192.168.1.2 remote-as 80000
```

Access lists that contain AS4s

AS4s that exist in access lists are displayed by the command, as shown.

```
device# show ip as-path-access-lists
ip as-path access list abc: 1 entries
  seq 10 permit _75000_
ip as-path access list def: 1 entries
  seq 5 permit _80000_
```

Formats of AS4s in show command output

To display the asdot and asdot+ notation for AS4s, enter the **as-format asdot** or **as-format asdot+** commands before you enter the **show ip bgp** command.

```
device# as-format asdot
device-mu2(config)# show ip bgp
Total number of BGP Routes: 1
Status codes: s suppressed, d damped, h history, * valid, > best, i internal, S stale
Origin codes: i - IGP, e - EGP, ? - incomplete
Network      Next Hop      Metric LocPrf Weight Path
*> 10.1.1.0/24 192.168.1.5      1      100      0      1.24464 100 200 655
5 1.0 1.1 1.2 1.3 1.9464 ?
```

Displaying BGP4 information

Syntax: as-format asdot

```
device# as-format asdot+
device# show ip bgp
Total number of BGP Routes: 1
Status codes: s suppressed, d damped, h history, * valid, > best, i internal, S stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network      Next Hop      Metric LocPrf Weight Path
*>  10.1.1.0/24   192.168.1.5    1      100      0      1.24464 0.100 0.200
0.65535 1.0 1.1 1.2 1.3 1.9464?
```

Syntax: as-format asdot+

Updating route information and resetting a neighbor session

The following sections describe how to update route information with a neighbor, reset a session with a neighbor, and close a session with a neighbor.

Any change to a policy (ACL, route map, and so on) is automatically applied to outbound routes that are learned from a BGP4 neighbor or peer group after the policy change occurs. However, you must reset the neighbor to update existing outbound routes.

Any change to a policy is automatically applied to inbound routes that are learned after the policy change occurs. However, to apply the changes to existing inbound routes (those inbound routes that were learned before the policy change), you must reset the neighbors to update the routes using one of the following methods:

- Request the complete BGP4 route table from the neighbor or peer group. You can use this method if the neighbor supports the refresh capability (RFCs 2842 and 2858). Most devices today support this capability.
- Clear (reset) the session with the neighbor or peer group. This is the only method you can use if soft reconfiguration is enabled for the neighbor.

You also can clear and reset the BGP4 routes that have been installed in the IP route table.

Using soft reconfiguration

The soft reconfiguration feature applies policy changes without resetting the BGP4 session. Soft reconfiguration does not request the neighbor or group to send the entire BGP4 table, nor does the feature reset the session with the neighbor or group. Instead, soft reconfiguration stores all the route updates received from the neighbor or group. When you request a soft reset of inbound routes, the software performs route selection by comparing the policies against the stored route updates, instead of requesting the neighbor BGP4 route table or resetting the session with the neighbor.

When you enable the soft reconfiguration feature, it sends a refresh message to the neighbor or group if the neighbor or group supports dynamic refresh. Otherwise, the feature resets the neighbor session. This step is required to ensure that the soft reconfiguration feature has a complete set of updates to use, and occurs only once, when you enable the feature. The feature accumulates all the route updates from the neighbor, eliminating the need for additional refreshes or resets when you change policies in the future.

To use soft reconfiguration:

- Enable the feature.
- Make the policy changes.
- Apply the changes by requesting a soft reset of the inbound updates from the neighbor or group.

Enabling soft reconfiguration

To configure a neighbor for soft reconfiguration, enter a command such as the following.

```
device(config-bgp) # neighbor 10.10.200.102 soft-reconfiguration inbound
```

This command enables soft reconfiguration for updates received from 10.10.200.102. The software dynamically resets the session with the neighbor, then retains all route updates from the neighbor following the reset.

Syntax: `[no] neighbor ip-addr | peer-group-name soft-reconfiguration inbound`

NOTE

The syntax related to soft reconfiguration is shown.

Placing a policy change into effect

To place policy changes into effect, enter a command such as the following.

```
device(config-bgp)# clear ip bgp neighbor 10.10.200.102 soft in
```

This command updates the routes by comparing the route policies against the route updates that the device has stored. The command does not request additional updates from the neighbor or otherwise affect the session with the neighbor.

Syntax: `clear ip bgp neighbor ip-addr | peer-group-name soft in`

NOTE

If you do not specify `in`, the command applies to both inbound and outbound updates.

NOTE

The syntax related to soft reconfiguration is shown.

Displaying the filtered routes received from the neighbor or peer group

When you enable soft reconfiguration, the device saves all updates received from the specified neighbor or peer group, including updates that contain routes that are filtered out by the BGP4 route policies in effect on the device. To display the routes that have been filtered out, enter the following command at any level of the CLI.

```
device# show ip bgp filtered-routes
Searching for matching routes, use ^C to quit...
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
      E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
      S:SUPPRESSED F:FILTERED s:STALE
Prefix      Next Hop      MED      LocPrf      Weight Status
1    10.3.0.0/8      192.168.4.106      100      0      EF
      AS_PATH: 65001 4355 701 80
2    10.4.0.0/8      192.168.4.106      100      0      EF
      AS_PATH: 65001 4355 1
3    10.60.212.0/22  192.168.4.106      100      0      EF
      AS_PATH: 65001 4355 701 1 189
```

The routes displayed are the routes that were filtered out by the BGP4 policies on the device. The device did not place the routes in the BGP4 route table, but did keep the updates. If a policy change causes these routes to be permitted, the device does not need to request the route information from the neighbor, but instead uses the information in the updates.

Syntax: `show ip bgp filtered-routes [ip-addr] [as-path-access-list num] [detail] [prefix-list string] [longer-prefixes]`

The `ip-addr` parameter specifies the IP address of the destination network.

The `as-path-access-list num` parameter specifies an AS-path ACL. Only the routes permitted by the AS-path ACL are displayed.

The `detail` parameter displays detailed information for the routes. (The example shows summary information.) You can specify any of the other options after `detail` to further refine the display request.

The `prefix-list string` parameter specifies an IP prefix list. Only routes permitted by the prefix list are displayed.

Displaying BGP4 information

If you also use the optional **longer-prefixes** parameter, then all statistics for routes that match the specified route or have a longer prefix than the specified route are displayed. For example, if you specify 10.157.0.0 longer, then all routes with the prefix 10.157 or that have a longer prefix (such as 10.157.22) are displayed.

Displaying all the routes received from the neighbor

To display all the route information received in route updates from a neighbor since you enabled soft reconfiguration, enter a command such as the following at any level of the CLI.

```
device# show ip bgp neighbor 192.168.4.106 routes
    There are 97345 received routes from neighbor 192.168.4.106
    Searching for matching routes, use ^C to quit...
    Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
           E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
           S:SUPPRESSED F:FILTERED s:STALE
    Prefix      Next Hop      MED      LocPrf      Weight      Status
  1    10.3.0.0/8      192.168.4.106      100      0      BE
      AS_PATH: 65001 4355 701 80
  2    10.4.0.0/8      192.168.4.106      100      0      BE
      AS_PATH: 65001 4355 1
  3    10.60.212.0/22  192.168.4.106      100      0      BE
      AS_PATH: 65001 4355 701 1 189
  4    10.6.0.0/8      192.168.4.106      0      0      BE

device# show ip bgp neighbor 192.168.4.106 routes
    There are 97345 received routes from neighbor 192.168.4.106
    Searching for matching routes, use ^C to quit...
    tatus A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
           E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH S:SUPPRESSED F:FILTEREDtatus A:AGGREGATE B:BEST b:NOT-
    INSTALLED-BEST C:CONFED_EBGP D:DAMPED
           E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH S:SUPPRESSED F:FILTERED
    Prefix      Next Hop      MED      LocPrf      Weight      Status
  1    10.3.0.0/8      192.168.4.106      100      0      BE
      AS_PATH: 65001 4355 701 8
  2    10.4.0.0/8      192.168.4.106      100      0      BE
      AS_PATH: 65001 4355 1
  3    10.60.212.0/22  192.168.4.106      100      0      BE
      AS_PATH: 65001 4355 701 1 189
  4    10.6.0.0/8      192.168.4.106      100      0      BE
```

Syntax: `show ip bgp neighbors ip-addr received-routes [detail]`

The **detail** parameter displays detailed information for the routes. This example shows summary information.

NOTE

The syntax for displaying received routes is shown. For complete command syntax, refer to [Displaying BGP4 neighbor information](#) on page 388.

Dynamically requesting a route refresh from a BGP4 neighbor

You can easily apply changes to filters that control BGP4 routes received from or advertised to a neighbor, without resetting the BGP4 session between the device and the neighbor. For example, if you add, change, or remove a BGP4 IP prefix list that denies specific routes received from a neighbor, you can apply the filter change by requesting a route refresh from the neighbor. If the neighbor also supports dynamic route refreshes, the neighbor resends its Adj-RIB-Out, its table of BGP4 routes. Using the route refresh feature, you do not need to reset the session with the neighbor.

The route refresh feature is based on the following specifications:

- RFC 2842. This RFC specifies the Capability Advertisement, which a BGP4 device uses to dynamically negotiate a capability with a neighbor.
- RFC 2858 for Multi-protocol Extension.

- RFC 2918, which describes the dynamic route refresh capability

The dynamic route refresh capability is enabled by default and cannot be disabled. When the device sends a BGP4 OPEN message to a neighbor, the device includes a Capability Advertisement to inform the neighbor that the device supports dynamic route refresh.

NOTE

The option for dynamically refreshing routes received from a neighbor requires the neighbor to support dynamic route refresh. If the neighbor does not support this feature, the option does not take effect and the software displays an error message. The option for dynamically re-advertising routes to a neighbor does not require the neighbor to support dynamic route refresh.

Dynamically refreshing routes

The following sections describe how to refresh BGP4 routes dynamically to put new or changed filters into effect.

To request a dynamic refresh of all routes from a neighbor, enter a command such as the following.

```
device(config-bgp-router)# clear ip bgp neighbor 192.168.1.170 soft in
```

This command asks the neighbor to send its BGP4 table (Adj-RIB-Out) again. The device applies its filters to the incoming routes and adds, modifies, or removes BGP4 routes as necessary.

Syntax: `clear ip bgp neighbor all | ip-addr | peer-group-name | as-num [soft-outbound | soft [in | out]]`

The **all**, **ip-addr**, **peer-group-name**, and **as-num** parameters specify the neighbor. The **ip-addr** parameter specifies a neighbor by its IP interface with the device. The **peer-group-name** specifies all neighbors in a specific peer group. The **as-num** parameter specifies all neighbors within the specified AS. The **all** parameter specifies all neighbors.

The **soft-outbound** parameter updates all outbound routes by applying the new or changed filters, but sends only the existing routes affected by the new or changed filters to the neighbor.

The **soft in** and **soft out** parameters specify whether you want to refresh the routes received from the neighbor or sent to the neighbor:

- **soft in** does one of the following:
 - If you enabled soft reconfiguration for the neighbor or peer group, **soft in** updates the routes by comparing the route policies against the route updates that the device has stored. Soft reconfiguration does not request additional updates from the neighbor or otherwise affect the session with the neighbor.
 - If you did not enable soft reconfiguration, **soft in** requests the entire BGP4 route table for the neighbor (Adj-RIB-Out), then applies the filters to add, change, or exclude routes.
 - If a neighbor does not support dynamic refresh, **soft in** resets the neighbor session.
- **soft out** updates all outbound routes, then sends the entire BGP4 router table for the device (Adj-RIB-Out) to the neighbor, after changing or excluding the routes affected by the filters.

If you do not specify **in** or **out**, the device performs both options.

NOTE

The **soft-outbound** parameter updates all outbound routes by applying the new or changed filters, but sends only the existing routes affected by the new or changed filters to the neighbor. The **soft out** parameter updates all outbound routes, then sends the entire BGP4 route table for the device (Adj-RIB-Out) to the neighbor, after changing or excluding the routes affected by the filters. Use **soft-outbound** if only the outbound policy is changed.

To dynamically resend all the device BGP4 routes to a neighbor, enter a command such as the following.

```
device(config-bgp)# clear ip bgp neighbor 192.168.1.170 soft out
```

This command applies filters for outgoing routes to the device BGP4 route table (Adj-RIB-Out), changes or excludes routes accordingly, then sends the resulting Adj-RIB-Out to the neighbor.

NOTE

The Brocade device does not automatically update outbound routes using a new or changed outbound policy or filter when a session with the neighbor goes up or down. Instead, the device applies a new or changed policy or filter when a route is placed in the outbound queue (Adj-RIB-Out). To place a new or changed outbound policy or filter into effect, you must enter a **clear ip bgp neighbor** command regardless of whether the neighbor session is up or down. You can enter the command without optional parameters or with the **soft out** or **soft-outbound** option. Either way, you must specify a parameter for the neighbor (*ip-addr, as-num, peer-group-name, or all*).

Displaying dynamic refresh information

You can use the **show ip bgp neighbors** command to display information for dynamic refresh requests. For each neighbor, the display lists the number of dynamic refresh requests the device has sent to or received from the neighbor and indicates whether the device received confirmation from the neighbor that the neighbor supports dynamic route refresh.

The RefreshCapability field indicates whether this device has received confirmation from the neighbor that the neighbor supports the dynamic refresh capability. The statistics in the Message Sent and Message Received rows under Refresh-Req indicate how many dynamic refreshes have been sent to and received from the neighbor. The statistic is cumulative across sessions.

```
device(config-bgp)# show ip bgp neighbor 10.4.0.2
1  IP Address: 10.4.0.2, AS: 5 (EBGP), RouterID: 100.0.0.1
   Description: neighbor 10.4.0.2
   State: ESTABLISHED, Time: 0h1m0s, KeepAliveTime: 0, HoldTime: 0
   PeerGroup: pgl
   Mutihop-EBGP: yes, ttl: 1
   RouteReflectorClient: yes
   SendCommunity: yes
   NextHopSelf: yes
   DefaultOriginate: yes (default sent)
   MaximumPrefixLimit: 90000
   RemovePrivateAs: : yes
   RefreshCapability: Received
Route Filter Policies:
  Distribute-list: (out) 20
  Filter-list: (in) 30
  Prefix-list: (in) pfl
  Route-map: (in) setnp1 (out) setnp2
Messages:
  Open      Update  KeepAlive  Notification  Refresh-Req
  Sent      : 1      1          1            0              0
  Received: 1      8          1            0              0
Last Update Time: NLRI      Withdraw      NLRI      Withdraw
                  Tx: 0h0m59s    ---          Rx: 0h0m59s    ---
Last Connection Reset Reason:Unknown
Notification Sent:      Unspecified
Notification Received: Unspecified
TCP Connection state: ESTABLISHED
Byte Sent: 115, Received: 492
Local host: 10.4.0.1, Local Port: 179
Remote host: 10.4.0.2, Remote Port: 8053
ISentSeq: 52837276 SendNext: 52837392 TotUnAck: 0
TotSent: 116 ReTrans: 0 UnAckSeq: 52837392
IRcvSeq: 2155052043 RcvNext: 2155052536 SendWnd: 16384
TotalRcv: 493 DupliRcv: 0 RcvWnd: 16384
SendQue: 0 RcvQue: 0 CngstWnd: 1460
```

Closing or resetting a neighbor session

You can close a neighbor session or resend route updates to a neighbor.

If you make changes to filters or route maps and the neighbor does not support dynamic route refresh, use the following methods to ensure that neighbors contain only the routes you want them to contain:

- If you close a neighbor session, the device and the neighbor clear all the routes they learned from each other. When the device and neighbor establish a new BGP4 session, they exchange route tables again. Use this method if you want the device to relearn routes from the neighbor and resend its own route table to the neighbor.
- If you use the soft-outbound option, the device compiles a list of all the routes it would normally send to the neighbor at the beginning of a session. However, before sending the updates, the device also applies the filters and route maps you have configured to the list of routes. If the filters or route maps result in changes to the list of routes, the device sends updates to advertise, change, or even withdraw routes on the neighbor as needed. This ensures that the neighbor receives only the routes you want it to contain. Even if the neighbor already contains a route learned from the device that you later decided to filter out, using the soft-outbound option removes that route from the neighbor.

You can specify a single neighbor or a peer group.

To close a neighbor session and thus flush all the routes exchanged by the device and the neighbor, enter the following command.

```
device# clear ip bgp neighbor all
```

Syntax: `clear ip bgp neighbor all` | *ip-addr* | *peer-group-name* | *as-num* [**soft-outbound** | **soft** [**in** | **out**]]

The **all**, *ip-addr*, *peer-group-name*, and *as-num* parameters specify the neighbor. The *ip-addr* parameter specifies a neighbor by its IP interface with the device. The *peer-group-name* specifies all neighbors in a specific peer group. The *as-num* parameter specifies all neighbors within an AS and has a range of 1 through 4294967295. The **all** keyword specifies all neighbors.

To resend routes to a neighbor without closing the neighbor session, enter a command such as the following.

```
device# clear ip bgp neighbor 10.0.0.1 soft out
```

Clearing and resetting BGP4 routes in the IP route table

To clear BGP4 routes from the IP route table and reset the routes, enter a command such as the following.

```
device# clear ip bgp routes
```

Syntax: `clear ip bgp routes` [*ip-addr/prefix-length*]

Clearing traffic counters

You can clear the counters (reset them to 0) for BGP4 messages.

To clear the BGP4 message counter for all neighbors, enter the following command.

```
device# clear ip bgp traffic
```

Syntax: `clear ip bgp traffic`

To clear the BGP4 message counter for a specific neighbor, enter a command such as the following.

```
device# clear ip bgp neighbor 10.0.0.1 traffic
```

To clear the BGP4 message counter for all neighbors within a peer group, enter a command such as the following.

```
device# clear ip bgp neighbor PeerGroup1 traffic
```

Syntax: `clear ip bgp neighbor all` | *ip-addr* | *peer-group-name* | *as-num* **traffic**

Clearing traffic counters

The **all** , *ip-addr* , *peer-group-name* , and *as-num* parameters specify the neighbor. The *ip-addr* parameter specifies a neighbor by its IP interface with the device. The *peer-group-name* specifies all neighbors in a specific peer group. The *as-num* parameter specifies all neighbors within the specified AS. The **all** parameter specifies all neighbors.

BGP4+

• BGP4+ overview.....	421
• BGP global mode	421
• IPv6 unicast address family.....	423
• BGP4+ neighbors.....	424
• BGP4+ peer groups.....	426
• Importing routes into BGP4+.....	428
• Advertising the default BGP4+ route.....	429
• Advertising the default BGP4+ route to a specific neighbor.....	429
• Using the IPv6 default route as a valid next hop for a BGP4+ route.....	430
• BGP4+ next hop recursion.....	430
• BGP4+ NLRIs and next hop attributes.....	431
• BGP4+ route reflection.....	432
• BGP4+ route aggregation.....	433
• BGP4+ multipath.....	434
• Route maps.....	435
• Redistributing prefixes into BGP4+.....	437
• Redistributing routes into BGP4+.....	437
• Specifying the weight added to BGP4+ received routes.....	438
• BGP4+ outbound route filtering.....	438
• BGP4+ confederations.....	440
• BGP4+ extended community.....	441
• BGP4+ graceful restart.....	443
• Generalized TTL Security Mechanism support.....	446
• Disabling the BGP AS_PATH check function.....	447
• Displaying BGP4+ statistics.....	448
• Displaying BGP4+ neighbor statistics.....	450

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)# ?
```

BGP global mode

Possible completions:

address-family	Enter Address Family command mode
address-filter	Configure IP address filters
aggregate-address	Configure BGP aggregate entries
always-compare-med	Allow comparing MED from different neighbors
always-propagate	Allow readvertisement of best BGP routes not in IP forwarding table
as-path-filter	Configure autonomous system path filters
as-path-ignore	Ignore AS_PATH length info for best route selection
bgp-redistribute-internal	Allow redistribution of iBGP routes into IGP
capability	Set capability
clear	Clear table/statistics/keys
client-to-client-reflection	Configure client to client route reflection
cluster-id	Configure Route-Reflector Cluster-ID
community-filter	Configure community list filters
compare-routerid	Compare router-id for identical BGP paths
confederation	Configure AS confederation parameters
dampening	Enable route-flap dampening
default-information-originate	
default-local-preference	Configure default local preference value
default-metric	Set metric of redistributed routes
distance	Define an administrative distance
enforce-first-as	Enforce the first AS for EBGp routes
fast-external-fallover	Reset session if link to EBGp peer goes down
graceful-restart	Enables the BGP graceful restart capability
local-as	Configure local AS number
maximum-paths	Forward packets over multiple paths
med-missing-as-worst	Consider routes missing MED attribute as least desirable
multipath	Enable multipath for ibgp or ebgp neighbors only
neighbor	Specify a neighbor router
network	Specify a network to announce via BGP
next-hop-enable-default	Enable default route for BGP next-hop lookup
next-hop-recursion	Perform next-hop recursive lookup for BGP route
readvertise	Allow readvertisement of best BGP routes not in IP forwarding table
redistribute	Redistribute information from another routing protocol
table-map	Map external entry attributes into routing table
timers	Adjust routing timers
update-time	Configure igp route update interval

The following neighbor configuration options are allowed under BGP global configuration mode:

```
device(config-bgp-rotuer)# neighbor 2001:2018:8192::125 ?
```

Possible completions:

activate	Allow exchange of route in the current family mode
advertisement-interval	Minimum interval between sending BGP routing updates
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
timers	BGP per neighbor timers
update-source	Source of routing updates

IPv6 unicast address family

The IPv6 unicast address family configuration level provides access to commands that allow you to configure BGP4+ unicast routes. The commands that you enter at this level apply only to the IPv6 unicast address family.

BGP4+ supports the IPv6 address family configuration level.

You can generate a configuration for BGP4+ unicast routes that is separate and distinct from configurations for IPv4 unicast routes.

The commands that you can access while at the IPv6 unicast address family configuration level are also available at the IPv4 unicast address family configuration levels. Each address family configuration level allows you to access commands that apply to that particular address family only.

Where relevant, this chapter discusses and provides IPv6-unicast-specific examples. You must first configure IPv6 unicast routing for any IPv6 routing protocol to be active.

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
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
advertisement-interval	Minimum interval between sending BGP routing updates
allowas-in	Accept as-path with my AS present in it
as-override	Override matching AS-number while sending update
capability	Advertise capability to the peer
default-originate	Originate default route to peer
description	Neighbor by description
ebgp-btsh	Enable EBGp TTL Security Hack Protection
ebgp-multihop	Allow EBGp neighbors not on directly connected networks
filter-list	Establish BGP filters
local-as	Assign local-as number to neighbor
maximum-prefix	Maximum number of prefix accept from this peer
next-hop-self	Disable the next hop calculation for this neighbor
password	Enable TCP-MD5 password protection
peer-group	Assign peer-group to neighbor
prefix-list	Prefix List for filtering routes

BGP4+ neighbors

<code>remote-as</code>	Specify a BGP neighbor
<code>remove-private-as</code>	Remove private AS number from outbound updates
<code>route-map</code>	Apply route map to neighbor
<code>route-reflector-client</code>	Configure a neighbor as Route Reflector client
<code>send-community</code>	Send community attribute to this neighbor
<code>shutdown</code>	Administratively shut down this neighbor
<code>soft-reconfiguration</code>	Per neighbor soft reconfiguration
<code>timers</code>	BGP per neighbor timers
<code>unsuppress-map</code>	Route-map to selectively unsuppress suppressed routes
<code>update-source</code>	Source of routing updates
<code>weight</code>	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 ipv6-address remote-as** command 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** command and specify the **ipv6** and **unicast** keywords to enter IPv6 address family configuration mode.

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


6. Enter the **neighbor ipv6-address 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 ipv6-address remote-as** command 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 ipv6-address update-source** command to specify an interface.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 update-source ethernet 1/3/1
```

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

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

7. Enter the **neighbor ipv6-address 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 ipv6-address route-map** command and specify the **out** keyword to apply a route map to outgoing routes.

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

9. Enter the **exit** command until you return to global configuration mode.

```
device(config-bgp-ipv6u)# exit
```

10. Enter the **route-map name permit** command to define the route map and enter route map configuration mode.

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

11. Enter the **set ipv6 next-hop** command and specify an IPv6 address to set the IPv6 address of the next hop.

```
device(config-routemap-myroumap)# 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 1/3/1
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 myroumap
device(config-bgp-ipv6u)# exit
device(config)# route-map myroumap permit 10
device(config-route-mapmyroumap)# 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-name peer-group** command to create a peer group.

```
device(config-bgp-router)# neighbor mypeergrout1 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 ipv6-address 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 ipv6-address peer-group** command to associate a neighbor with the peer group.

```
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group mypeergroup1
```

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

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

9. Enter the **neighbor peer-group-name 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-name peer-group** command to create a peer group.

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

6. Enter the **neighbor ipv6-address peer-group** command to associate a neighbor with the peer group.

```
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group p1
```

Importing routes into BGP4+

7. Enter the **neighbor ip address peer-group** command to associate a neighbor with the peer group.

```
device(config-bgp-router)# neighbor 10.0.0.1 peer-group p1
```

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

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

9. Enter the **neighbor peer-group-name 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
```

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 ipv6-address remote-as** command 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** command and specify the **ipv6** and **unicast** keywords 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 BGP 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** command and specify the **ipv6** and **unicast** keywords to enter IPv6 address family configuration mode.

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

Using the IPv6 default route as a valid next hop for a BGP4+ route

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** command and specify the **ipv6** and **unicast** keywords 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 IGP does 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** command and specify the **ipv6** and **unicast** keywords 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** command and specify the **ipv6** and **unicast** keywords to enter IPv6 address family configuration mode.

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

5. Enter the **neighbor ipv6-address route-reflector-client** command 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** command and specify the **ipv6** and **unicast** keywords 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** command and specify the **ipv6** and **unicast** keywords 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
```

The following example sets the maximum number of BGP4+ shared paths to that of the value already configured using the **ip load-sharing** command.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# maximum-paths use-load-sharing
```

Route maps

Route maps must be applied to IPv6 unicast address prefixes in IPv6 address family configuration mode.

By default, route maps that are applied under IPv4 address family configuration mode using the **neighbor route-map** command are applied to only IPv4 unicast address prefixes. To apply route maps to IPv6 unicast address prefixes, the **neighbor route-map** command must be used in IPv6 address family configuration mode. The route maps are applied as the inbound or outbound routing policy for neighbors under the specified address family. Configuring separate route maps under each address family type simplifies managing complicated or different policies for each address family.

Configuring a route map for BGP4+ prefixes

Route maps can be applied to IPv6 unicast address prefixes either as the inbound or outbound routing policy for neighbors under the specified address family.

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

```
device# configure terminal
```

2. Enter the **ipv6 prefix-list** command and enter a name to configure an IPv6 prefix list.

```
device(config)# ipv6 prefix-list myprefixlist seq 10 permit 2001:db8::/32
```

The prefix list name, sequence number, and permits packets are specified.

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

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

4. Enter the **match ipv6 address** command and specify the name of a prefix list.

```
device(config-route-map-myroutemap)# match ipv6 address prefix-list myprefixlist
```

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

```
device(config-route-map-myroutemap)# 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 ipv6-address remote-as** command 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** command and specify the **ipv6** and **unicast** keywords to enter IPv6 address family configuration mode.

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

10. Enter the **neighbor ipv6-address 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 ipv6-address route-map** command and specify the **out** keyword to apply a route map to outgoing routes.

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

The following example applies a route map, "myroutemap", as the outbound routing policy for a neighbor.

```
device# configure terminal
device(config)# ipv6 prefix-list myprefixlist seq 10 permit 2001:db8::/32
device(config)# route-map myroutemap permit 10
device(config-route-map-myroutemap)# match ipv6 address prefix-list myprefixlist
device(config-route-map-myroutemap)# exit
device(config)# router bgp
device(config-bgp-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 RIPng routes can be redistributed into BGP. This task redistributes RIPng 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 unicast** command and specify the **ipv6** and **unicast** keywords to enter IPv6 address family configuration mode.

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

4. Enter the **redistribute** command using the **rip** keyword to redistribute IPv6 RIP routes.

```
device(config-bgp-ipv6u)# redistribute rip
```

The following example redistributes RIPng prefixes into BGP4+.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# redistribute rip
```

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 BGP IPv6 address family unicast configuration mode.

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

Specifying the weight added to BGP4+ received routes

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.

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
```

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** command and specify the **ipv6** and **unicast** keywords to enter IPv6 address family configuration mode.

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

4. Enter the **neighbor ipv6-address activate** command to add a neighbor.

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

5. Enter the **neighbor ipv6-address prefix-list** command and specify 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 a community ACL

A BGP community ACL can be configured, and BGP community attributes set in a route map instance.

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

```
device# configure terminal
```

2. Enter the **ip community-list extended** command using the **permit** keyword to configure a BGP community ACL.

```
device(config)# ip community-list extended 1 permit ^[1-2]23
```

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

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

4. Enter the **match community** command and specify a community list name.

```
device(config-route-map-ComRmap)# match community 1
```

5. Enter the **set community** command to set the BGP community attributes.

```
device(config-route-map-ComRmap)# set community 323:1 additive
```

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

```
device(config-route-map-ComRmap)# exit
```

BGP4+ extended community

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

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

8. Enter the **set community** command to set the BGP community attributes.

```
device(config-route-map-sendComRmap)# set community 3:3
```

The following example configures a BGP community ACL and sets the BGP community attributes in a route map instance.

```
device# configure terminal
device(config)# ip community-list extended 1 permit ^[1-2]23
device(config)# route-map ComRmap permit 10
device(config-route-map-ComRmap)# match community 1
device(config-route-map-ComRmap)# set community 323:1 additive
device(config-route-map-ComRmap)# exit
device(config)# route-map sendComRmap permit 10
device(config-route-map-sendComRmap)# set community 3:3
```

Applying a BGP extended community filter

A BGP extended community filter can be applied.

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

```
device# configure terminal
```

2. Enter the **ip community-list extended** command using the **permit** keyword to configure a BGP community ACL.

```
device(config)# ip community-list extended 1 permit ^[1-2]23
```

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

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

4. Enter the **match community** command and specify a community list name.

```
device(config-route-map-ComRmap)# match community 1
```

5. Enter the **set local-preference** command and specify a value to set a BGP local-preference path attribute.

```
device(config-route-map-ComRmap)# set local-preference 200
```

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 ipv6-address remote-as** command 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 **neighbor ipv6-address update-source** command to specify an interface.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 update-source ve 1000
```

10. Enter the **address-family** command and specify the **ipv6** and **unicast** keywords to enter IPv6 address family configuration mode.

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

11. Enter the **neighbor ipv6-address activate** command to enable the exchange of information with the neighbor.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
```

12. Enter the **neighbor ipv6-address 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 ComRmap
```

13. Enter the **neighbor ipv6-address send-community** command 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
```

The following example applies a BGP extended community filter.

```
device# configure terminal
device(config)# ip community-list extended 1 permit ^[1-2]23
device(config)# route-map ComRmap permit 10
device(config-route-map-ComRmap)# match community 1
device(config-route-map-ComRmap)# set local-preference 200
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 ve 1000
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 ComRmap
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 send-community
```

BGP4+ graceful restart

BGP4+ graceful restart (GR) allows for restarts where 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 and peers when it is performing a restart.

When a BGP session is established, GR capability for BGP is negotiated by neighbors and peers 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 enabled by default in both IPv4 and IPv6 address families.

NOTE

BGP4+ GR can be configured for a global routing instance or for a specified VRF instance.

NOTE

BGP4+ GR is supported in ICX switches in a stack.

Disabling BGP4+ graceful restart

The BGP4+ graceful restart (GR) feature 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 bgp** command to enable BGP routing.

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

3. (Optional) Enter the **address-family** command and specify the **ipv6** and **unicast** keywords to enter IPv6 address family configuration mode.

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

4. Enter the **no graceful restart** command to disable graceful restart at the IPv6 address family configuration level.

```
device(config-bgp-ipv6u)# no graceful-restart
```

In the following example, the graceful restart feature is disabled at the IPv6 address family configuration level.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv6u)# no graceful-restart
```

Re-enabling BGP4+ graceful restart

If the BGP4+ graceful restart (GR) feature is disabled on a routing device, it can be re-enabled, providing it with the capability to inform its neighbors and peers when it is performing a restart.

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 ipv6-address remote-as** command 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** command and specify the **ipv6** and **unicast** keywords to enter IPv6 address family configuration mode.

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

6. Enter the **neighbor ipv6-address 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 re-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 re-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 re-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 re-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.

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.

When GTSM protection is enabled, BGP control packets sent by the device to a neighbor have a Time To Live (TTL) value of 255. In addition, the device expects the BGP control packets received from the neighbor to have a TTL value of either 254 or 255. For multihop peers, the device expects the TTL for BGP control packets received from the neighbor to be greater than or equal to 255, minus the configured number of hops to the neighbor. If the BGP control packets received from the neighbor do not have the anticipated value, the device drops them.

For more information on GTSM protection, refer to RFC 3682.

Assumptions and limitations

- GTSM is supported for both directly connected peering sessions and multihop eBGP peering sessions.
- GTSM is supported for eBGP only.
- GTSM does not protect the integrity of data sent between eBGP peers and does not validate eBGP peers through any authentication method.
- GTSM validates only the locally configured TTL count against the TTL field in the IP packet header.
- GTSM should be configured on each participating device to maximize the effectiveness of this feature.
- When GTSM is enabled, the eBGP session is secured in the incoming direction only and has no effect on outgoing IP packets or the remote device.

Configuring GTSM for BGP4+

Generalized TTL Security Mechanism (GTSM) can be configured to protect external Border Gateway Protocol (eBGP) peering sessions .

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

```
device# configure terminal
```

2. Enter the **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 BGP IPv6 address family unicast configuration mode.

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

5. Enter the **neighbor ipv6-address remote-as** command to add a neighbor.

```
device(config-bgp-ipv6)# neighbor 2001:2018:8192::125 remote-as 2
```

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

```
device(config-bgp-ipv6)# 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)# address-family ipv6 unicast
device(config-bgp-ipv6)# neighbor 2001:2018:8192::125 remote-as 2
device(config-bgp-ipv6)# 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 BGP IPv6 address family unicast configuration mode.

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

4. Enter the **neighbor ipv6-address 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-ipv6)# 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-ipv6)# 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: 113.1.1.1   Local AS Number: 65020
Confederation Identifier: not configured
Confederation Peers:
Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
Number of Neighbors Configured: 2, UP: 1
Number of Routes Installed: 5, Uses 430 bytes
Number of Routes Advertising to All Neighbors: 7 (7 entries), Uses 336 bytes
Number of Attribute Entries Installed: 4, Uses 360 bytes
Neighbor Address      AS#   State  Time      Rt:Accepted  Filtered  Sent  ToSend
2001:db8:113:113::2   65001 CONN   1d14h32m    0           0       0     4
2001:db8:400:400::2   65020 ESTAB   3h59m24s    2           0       3     0
```

This example output gives summarized BGP4+ information.

2. Enter the **show ipv6 bgp attribute-entries** command.

```
device> show ipv6 bgp attribute-entries

Total number of BGP Attribute Entries: 4
1      Next Hop   : ::                                MED :1
      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: 1, Source As 0
      Address: 0x2a8bd092 Hash:364 (0x10000000)
      Links: 0x0, 0x0
      Reference Counts: 2:0:4, Magic: 3
```

This example shows information about two route-attribute entries that are stored in device memory.

3. Enter the **show ipv6 bgp peer-group** command.

```
device> show ipv6 bgp peer-group peer_group1

1  BGP peer-group is peer_group1
    Address family : IPV4 Unicast
        no activate
    Address family : IPV4 Multicast
        no activate
    Address family : IPV6 Unicast
        activate
    Address family : IPV6 Multicast
        no activate
    Address family : VPNV4 Unicast
        no activate
    Address family : L2VPN VPLS
        no activate
    Members:
        IP Address: 2000:400:400:400::3, AS: 65020
```

This example shows output for a peer group called "peer_group1".

4. Enter the **show ipv6 bgp routes** command.

```
device> show ipv6 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  2001:db8:10:10::/64 ::          1              100    32768  BL
   AS_PATH:
2  2001:db8:113:113::/64 ::          1              100    32768  BL
   AS_PATH:
3  2001::db8:400::/64  ::          0          100    32768  BL
   AS_PATH:
4  2001:db8:400:400::/64 2001:db8:400:400::2
                                0          400      0      I
   AS_PATH: 65005 65010
```

This example shows general BGP4+ route information.

5. Enter the **show ipv6 bgp routes** command, using the **detail** keyword.

```
device> show ipv6 bgp routes detail

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
1 Prefix: 2001:db8:10:10::/64, Status: BL, Age: 8h31m39s
  NEXT_HOP: ::, Learned from Peer: Local Router
  LOCAL_PREF: 100, MED: 0, ORIGIN: incomplete, Weight: 32768
  AS_PATH:
  Adj_RIB_out count: 3, Admin distance 1
2 Prefix: 2001:db8:113:113::/64, Status: BL, Age: 6h58m35s
  NEXT_HOP: ::, Learned from Peer: Local Router
  LOCAL_PREF: 100, MED: 0, ORIGIN: igp, Weight: 32768
  AS_PATH:
  Adj_RIB_out count: 3, Admin distance 1
3 Prefix: 2001:db8:202:202::/64, Status: BI, Age: 5h42m36s
  NEXT_HOP: 2001:db8:400:400::2, Metric: 0, Learned from Peer: 2001:db8:400:400::2 (65020)
  LOCAL_PREF: 400, MED: 0, ORIGIN: incomplete, Weight: 0
  AS_PATH: 65005 65010
  Adj_RIB_out count: 1, Admin distance 200
4 Prefix: 2001:db8:400:400::/64, Status: BL, Age: 5h43m14s
  NEXT_HOP: ::, Learned from Peer: Local Router
  LOCAL_PREF: 100, MED: 0, ORIGIN: igp, Weight: 32768
  AS_PATH:
  Adj_RIB_out count: 3, Admin distance 1
```

This example shows detailed BGP4+ route information.

Displaying BGP4+ neighbor statistics

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: 2
1  IP Address: 2001:1001::1, AS: 63753 (IBGP), RouterID: 1.0.0.1, VRF: default-vrf
   Description: SWD-2
   State: ESTABLISHED, Time: 0h47m50s, KeepAliveTime: 60, HoldTime: 180
   KeepAliveTimer Expire in 26 seconds, HoldTimer Expire in 168 seconds
   Minimal Route Advertisement Interval: 0 seconds
   MD5 Password: $Qj0tZHMlXC1vbJYt
   UpdateSource: Loopback 1
   NextHopSelf: yes
   RefreshCapability: Received
   GracefulRestartCapability: Received
     Restart Time 120 sec, Restart bit 0
     afi/safi 2/1, Forwarding bit 0
   GracefulRestartCapability: Sent
     Restart Time 120 sec, Restart bit 0
     afi/safi 2/1, Forwarding bit 0
   Messages:   Open      Update  KeepAlive Notification Refresh-Req
...
```

This example output gives summarized information about BGP4+ neighbors.

2. Enter the **show ipv6 bgp neighbors advertised-routes** command.

```
device. show ipv6 bgp neighbors 2001:db8::110 advertised-routes

There are 2 routes advertised to neighbor 2001:db8::110
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
      Prefix      Next Hop    MED LocPrf  Weight Status
1      2001:db8::/32      ::          1          32768  BL
      AS_PATH:
2      2001:db8::/16      ::          1          32768  BL
      AS_PATH:
```

This example shows information about all the routes the BGP4+ networking device advertised to the neighbor.

3. Enter the **show ipv6 bgp neighbors received-routes** command.

```
device> show ipv6 bgp neighbors 2001:db8:400:400::2 received-routes

There are 4 received routes from neighbor 2001:db8:400:400::2
Searching for matching routes, use ^C to quit...
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      2001:db8:202:202::/64  2001:db8:400:400::2  0    400    0      BI
      AS_PATH: 65005 65010
2      2001:db8:400:400::/64  2001:db8:400:400::2  0    400    0      I
      AS_PATH: 65005 65010
```

This example lists all route information received in route updates from BGP4+ neighbors of the device since the soft-reconfiguration feature was enabled.

4. Enter the **show ipv6 bgp neighbors rib-out-routes** command.

```
device> show ipv6 bgp neighbors 2001:db8::110 rib-out-routes

      There are 2 RIB_out routes for neighbor 2001:db8::110
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
Prefix      Next Hop      Metric      LocPrf      Weight Status
1    2001:db8::/32          ::           1          100      32768   BL
    AS_PATH:
2    2001:db8::/16          ::           1          100      32768   BL
    AS_PATH:
```

This example shows information about BGP4+ outbound RIB routes.

VRRPv2

• VRRPv2 overview.....	453
• Enabling an owner VRRP device.....	458
• Enabling a backup VRRP device.....	460
• Configuring simple text authentication on VRRP interfaces.....	461
• Configuring MD5 authentication on VRRP interfaces.....	462
• Abdicating VRRP master device status.....	463
• Tracked ports and track priority with VRRP and VRRP-E.....	465
• VRRP backup preemption.....	466
• Accept mode for backup VRRP devices.....	467
• Suppressing RIP route advertisements on VRRP backup devices.....	469
• VRRP-Ev2 overview.....	469
• Enabling a VRRP-E device.....	470
• VRRP-E load-balancing using short-path forwarding.....	471
• VRRP-E hitless upgrade.....	474
• VRRP-E slow start timer.....	476
• Configuration example: ISSU upgrade using VRRP-E.....	477
• Displaying VRRPv2 information.....	479
• Clearing VRRPv2 statistics.....	480

VRRPv2 overview

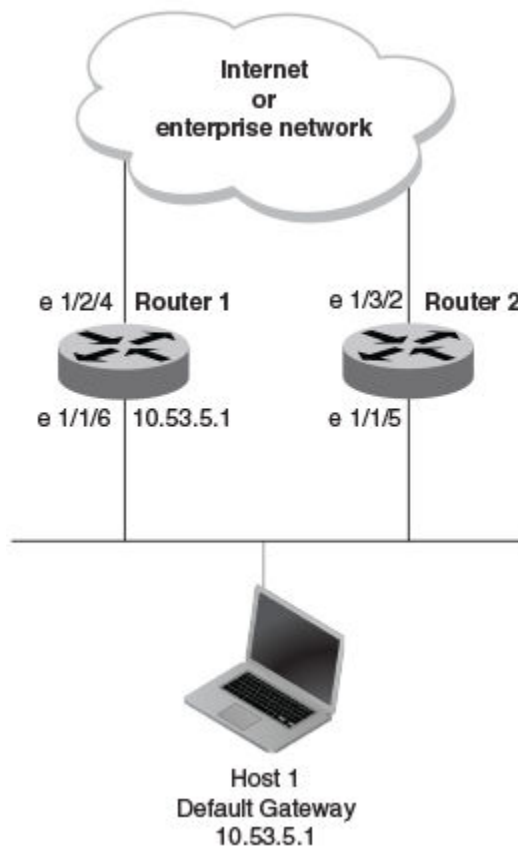
Virtual Router Redundancy Protocol (VRRP) is an election protocol that provides redundancy to routers within a Local Area Network (LAN).

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. Brocade developed VRRP-E as a proprietary protocol to address some limitations in standards-based VRRP.

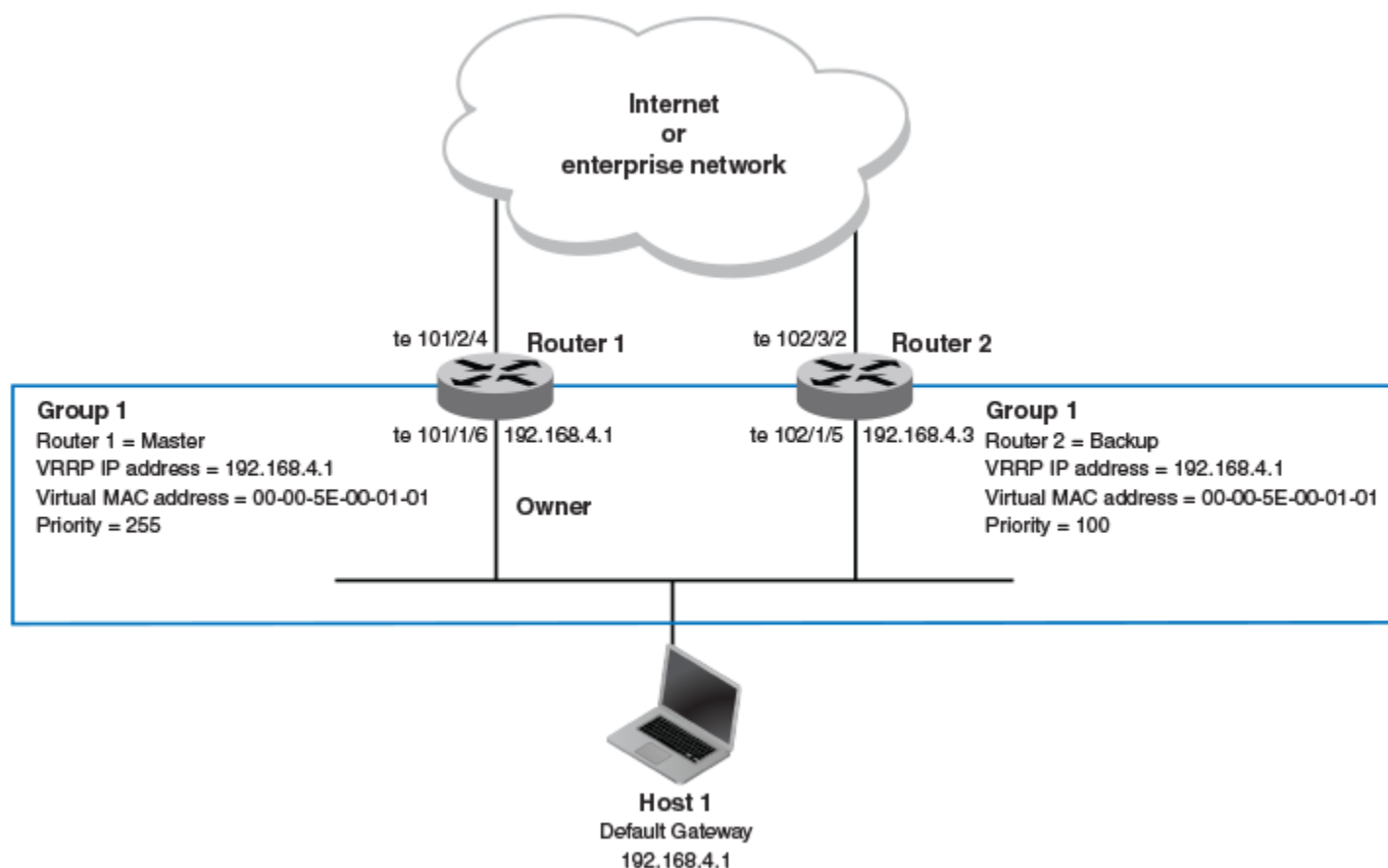
Before examining more details about how VRRP works, it is useful to see why VRRP was developed to solve the issue of a single point of failure.

FIGURE 36 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 10.53.5.1 on Router 1 as its default gateway. If this interface goes down, Host1 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 37 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 RIP. Their operation is unaffected when VRRP is enabled on the same interface as BGP4, OSPF, and RIP.

VRRP terminology

The following are key terms and definitions that you must understand before implementing VRRP in your network.

The following VRRP-related terms are in logical order, not alphabetic order:

Virtual router

A collection of physical routers that can use VRRP to provide redundancy to routers within a LAN.

Virtual router ID

A group of physical routers that are assigned to the same virtual router ID (VRID).

Virtual router address

The virtual router IP address must belong to the same subnet as a real IP address configured on the VRRP interface, and it can be the same as a real IP address configured on the VRRP interface. The virtual router whose virtual IP address is the same as a real IP address is the IP address owner and the default master.

Owner

The owner is the physical router whose real interface IP address is the IP address that you assign to the virtual router. The owner responds to packets addressed to any of the IP addresses in the corresponding virtual router. The owner, by default, is the master and has the highest priority (255).

Master

The physical router that responds to packets addressed to any of the IP addresses in the corresponding virtual router. For VRRP, if the physical router whose real interface IP address is the IP address of the virtual router, then this physical router is always the master.

Backup

Routers that belong to a virtual router, but are not the master. If the master becomes unavailable, the backup router with the highest priority (a configurable value) becomes the new master. By default, routers are given a priority of 100.

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 interval,

the backup devices assume that the master device is offline. When the master device is offline, the backup device with the highest priority assumes the role of the master device.

NOTE

The hello intervals must be set to the same value on both owner and backup devices for the same VRID.

dead interval

The dead interval is defined as the period of time for which backup devices wait for a hello message from the master device before assuming that the master device is offline. An immediate switchover to the backup device with the highest priority is triggered after the dead interval expires and there is no hello message from the master device. If a value for the dead interval is not configured, the default value is calculated as three times the hello interval plus the skew time. Skew time is defined as $(256 - \text{priority})/256$.

NOTE

The dead interval must be set to the same value on both owner and backup devices for the same VRID.

backup hello message state and interval

By default, backup devices do not send hello messages to advertise themselves to the master device. Hello messages from backup devices can be activated, and the messages are sent at 60-second intervals, by default. The interval between the backup hello messages can be modified.

VRRP authentication

The VRRP authentication type is not a parameter specific to the virtual router ID (VRID). VRRP uses the authentication type associated with the interfaces on which the virtual router ID (VRID) is defined.

If your interfaces do not use authentication, neither does VRRP. For example, if you configure your device interfaces to use a simple password to authenticate traffic, VRRP uses the same simple password, and VRRP packets that do not contain the password are dropped.

In summary, if the interfaces on which you configure the virtual router ID use authentication, the VRRP or 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.
- Simple—The interfaces use a simple text string as a password in packets that they send. If the interfaces use simple password authentication, the virtual router configured on the interfaces must use the same authentication type and the same password.
- MD5—This method of authentication ensures that the packet is authentic and cannot be modified in transit. Syslog and SNMP traps are generated when a packet is dropped due to MD5 authentication failure. MD5 authentication is supported only in VRRP-E, and the device configuration is unique on a per-interface basis. The MD5 authentication configuration on an interface takes effect for all VRRP-E virtual routers configured on a particular interface.

NOTE

Using MD5 authentication implies that the software does not need to run checksum verification on the receiving device because the authentication code (the message digest 5 algorithm) is used to verify the integrity of the VRRP-E message header.

NOTE

Authentication is not supported for VRRPv3.

VRRP master device abdication to backup device

To allow temporary control of a VRRP virtual router ID (VRID) to pass to a backup device, you can force the master device to abdicate to a backup device by setting a lower priority.

Changing the priority of a VRRP master device allows a temporary abdication of the master device status to allow a backup device with a higher priority to assume the master device role. By default, a VRRP owner device has a priority of 255, and the lower priority must be set to a lower priority than at least one of the backup devices associated with the VRID.

When you change the priority of a VRRP owner, the change takes effect only for the current power cycle. The change is not saved to the startup configuration file when you save the configuration, and it is not retained across a reload or reboot. Following a reload or reboot, the VRRP owner again has priority 255.

NOTE

This feature supports IPv4 VRRP only. IPv6 VRRP, VRRP-E, and IPv6 VRRP-E are not supported.

ARP and VRRP control packets

Control packets for ARP and VRRP are handled differently by VRRP and VRRP-E.

Source MAC addresses in VRRP control packets

- VRRP—The virtual MAC address is the source.
- VRRP-E—The physical MAC address is the source.

VRRP control packets

- VRRP—Control packets are IP type 112 (reserved for VRRP), and they are sent to the VRRP multicast address 224.0.0.18.
- VRRP-E—Control packets are UDP packets destined to port 8888, and they are sent to the all-router multicast address 224.0.0.2.

Gratuitous ARP

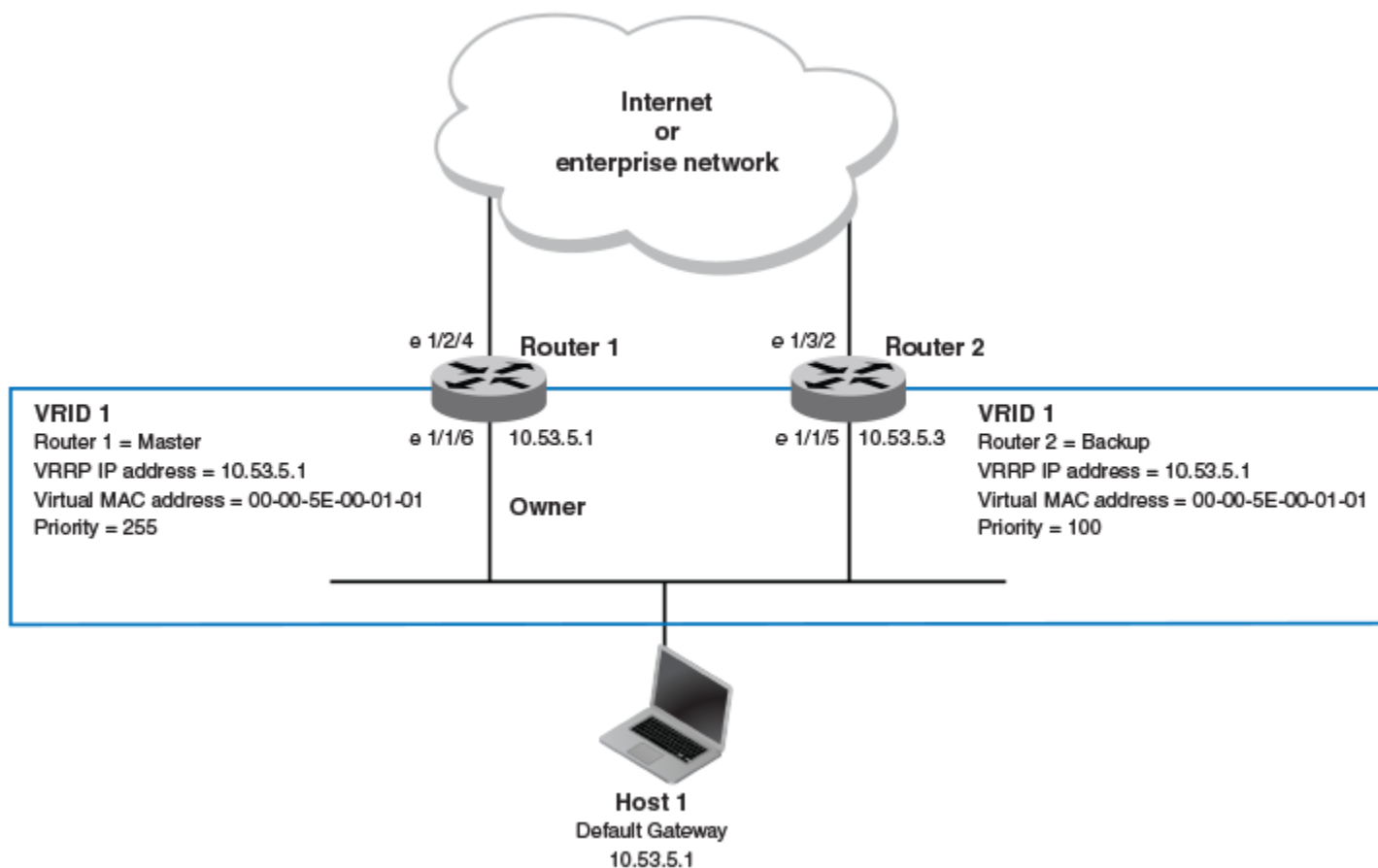
When a VRRP device (either master or backup) sends an ARP request or a reply packet, the MAC address of the sender is the MAC address of the router interface. One exception is if the owner sends an ARP request or a reply packet, in which case the MAC address of the sender is the virtual MAC address. Only the master answers an ARP request for the virtual router IP address. Any backup router that receives this request forwards the request to the master.

- VRRP—A control message is sent only once when the VRRP device assumes the role of the master VRRP device.
- VRRP-E—A control message is sent every 2 seconds by the VRRP-E master device because VRRP-E control packets do not use the virtual MAC address.

Enabling an owner VRRP device

This task is performed on the device that is designated as the owner VRRP device because the IP address of one of its physical interfaces is assigned as the IP address of the virtual router. For example, Router 1 is the owner VRRP device in the figure that follows. For each VRRP session, there are master and backup routers, and the owner router is elected, by default, as the master router.

FIGURE 38 Basic VRRP topology



1. On the device designated as the owner VRRP device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP.

```
device(config)# router vrrp
```

3. Configure the Ethernet interface link for Router 1.

```
device(config)# interface ethernet 1/1/6
```

4. Configure the IP address of the interface.

```
device(config-if-e1000-1/1/6)# ip address 10.53.5.1/24
```

5. Assign Router 1 to the virtual router ID (VRID) 1.

```
device(config-if-e1000-1/1/6)# ip vrrp vrid 1
```

NOTE

You can assign a VRID number in the range of 1 through 255.

Enabling a backup VRRP device

- Designate this router as the VRRP owner device.

```
device(config-if-e1000-1/1/6-vrid-1)# owner
```

- Configure the VRRP version.

```
device(config-if-e1000-1/1/6-vrid-1)# version 2
```

- Configure the IP address of the VRID.

```
device(config-if-e1000-1/1/6-vrid-1)# ip-address 10.53.5.1
```

- Enable the VRRP session.

```
device(config-if-e1000-1/1/6-vrid-1)# activate
```

The following example configures a VRRP owner device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/1/6
device(config-if-e1000-1/1/6)# ip address 10.53.5.1/24
device(config-if-e1000-1/1/6)# ip vrrp vrid 1
device(config-if-e1000-1/1/6-vrid-1)# owner
device(config-if-e1000-1/1/6-vrid-1)# version 2
device(config-if-e1000-1/1/6-vrid-1)# ip-address 10.53.5.1
device(config-if-e1000-1/1/6-vrid-1)# activate
VRRP router 1 for this interface is activating
```

Enabling a backup VRRP device

This task is performed on any device that is designated as a backup VRRP device. For each VRRP virtual routing instance, there is one master device and all other devices are backups. For example, Router 2 in [Figure 38](#) on page 459 is assigned as a backup device.

Repeat this task for all devices that are to be designated as backup devices.

- On the device designated as a backup VRRP device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

- Globally enable VRRP.

```
device(config)# router vrrp
```

- Configure the Ethernet interface link.

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

- Configure the IP address of the interface for Router 2. All devices configured for the same virtual router ID (VRID) must be on the same subnet.

```
device(config-if-e1000-1/1/5)# ip address 10.53.5.3/24
```

- Assign Router 2 to VRID 1, the same VRID as Router 1.

```
device(config-if-e1000-1/1/5)# ip vrrp vrid 1
```

NOTE

You can assign a VRID number in the range of 1 through 255.

- Designate this router as a backup VRRP device.

```
device(config-if-e1000-1/1/5-vrid-1)# backup priority 110
```

While configuring a backup device, you can set a priority that is used when a master VRRP device goes offline. The backup device with the highest priority will assume the role of master device.

- Configure the VRRP version.

```
device(config-if-e1000-1/1/5-vrid-1)# version 2
```

- Configure the number of seconds between hello messages.

```
device(config-if-e1000-1/1/5-vrid-1)# hello-interval 10
```

- By default, backup VRRP devices do not send hello messages to advertise themselves to the master. Use the following command to enable a backup router to send hello messages to the master VRRP device.

```
device(config-if-e1000-1/1/5-vrid-1)# advertise backup
```

- Configure the IP address of the VRID.

```
device(config-if-e1000-1/1/5-vrid-1)# ip-address 10.53.5.1
```

The VRID IP address is the same virtual IP address you used for Router 1.

- Enable the VRRP session.

```
device(config-if-e1000-1/1/5-vrid-1)# activate
VRRP router 1 for this interface is activating
```

The following example configures a VRRP backup device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/1/5
device(config-if-e1000-1/1/5)# ip address 10.53.5.3/24
device(config-if-e1000-1/1/5)# ip vrrp vrid 1
device(config-if-e1000-1/1/5-vrid-1)# backup priority 110
device(config-if-e1000-1/1/5-vrid-1)# version 2
device(config-if-e1000-1/1/5-vrid-1)# hello-interval 10
device(config-if-e1000-1/1/5-vrid-1)# advertise backup
device(config-if-e1000-1/1/5-vrid-1)# ip-address 10.53.5.1
device(config-if-e1000-1/1/5-vrid-1)# activate
VRRP router 1 for this interface is activating
```

Configuring simple text authentication on VRRP interfaces

A simple text password can be used for interface authentication in a network. VRRP uses the authentication type associated with the interfaces on which you define the virtual router ID (VRID).

A VRRP session must be configured and running.

If you configure your device interfaces to use a simple password to authenticate traffic, VRRP interfaces can be configured with the same simple password, and VRRP packets that do not contain the password are dropped. If your interfaces do not use authentication, neither does VRRP. Repeat this task on all interfaces on all devices that support the VRID.

NOTE

This task supports VRRPv2 and VRRP-Ev2 only. VRRPv3 and VRRP-Ev3 are not supported.

1. From privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP.

```
device(config)# router vrrp
```

3. Configure an Ethernet interface.

```
device(config)# interface ethernet 1/1/6
```

4. Enter the simple text password configuration using the **ip vrrp auth-type** command with a text password.

```
device(config-if-e1000-1/1/6)# ip vrrp auth-type simple-text-auth yourpwd
```

5. Verify the password on the interface using the **show ip vrrp** command with either the VRID or Ethernet options.

```
device(config-if-e1000-1/1/6-vrid-1)# show ip vrrp
```

```
Total number of VRRP routers defined: 1
Interface ethernet 1/1/6
auth-type simple text authentication
VRID 1
state backup
administrative-status enabled
mode owner
priority 99
current priority 99
hello-interval 1 sec
ip-address 10.53.5.1
backup routers 10.53.5.2
```

In this example, the authentication type is simple text authentication. A **show running-config** command with appropriate parameters will actually display the password. The output verifies the type of authentication.

Configuring MD5 authentication on VRRP interfaces

Interfaces can be configured with an MD5 encrypted password for authentication, and VRRP can use the same authentication type associated with the interfaces on which you define the virtual router ID (VRID).

If you configure your device interfaces to use an MD5 encrypted password to authenticate traffic, VRRP interfaces can be configured with the same MD5 password, and VRRP packets that do not contain the password are dropped. If your interfaces do not use authentication, neither does VRRP. Repeat this task on all interfaces on all devices that support the VRID.

1. From privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP.

```
device(config)# router vrrp
```

3. Specify an interface associated with the VRRP VRID.

```
device(config)# interface ethernet 1/1/6
```

4. Enter the MD5 password configuration using the **ip vrrp auth-type** command with a text password. The password will be encrypted when saved in the configuration file. When an MD5 authentication password is configured on an interface, a syslog message is displayed.

```
device(config-if-e1000-1/1/6)# ip vrrp auth-type md5-auth gy42mb
```

5. Verify the password on the interface using the **show ip vrrp** command.

```
device(config-if-e1000-1/1/6-vrid-1)# show ip vrrp
```

```
Total number of VRRP routers defined: 1
Interface ethernet 1/1/6
auth-type MD5 authentication
VRID 1
state backup
administrative-status enabled
mode owner
priority 99
current priority 99
hello-interval 1 sec
ip-address 10.53.5.1
backup routers 10.53.5.2
```

In this example, the auth-type is MD5 authentication where the entered password is encrypted. A **show run** command with appropriate parameters will actually display the encrypted password, and you can use the **enable password-display** command to actually display the encrypted password. The output verifies the type of authentication.

The following example enables MD5 authentication on Ethernet interface 1/1/6 and verifies the authentication type.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/1/6
device(config-if-e1000-1/1/6)# ip vrrp auth-type MD5 yourpwd
device(config-if-e1000-1/1/6-vrid-1)# show ip vrrp
```

```
Total number of VRRP routers defined: 1
Interface ethernet 1/1/6
auth-type MD5 authentication
VRID 1
state backup
administrative-status enabled
mode owner
priority 99
current priority 99
hello-interval 1 sec
ip-address 10.53.5.1
backup routers 10.53.5.2
```

Abdicating VRRP master device status

Changing the priority of a VRRP master device allows a temporary abdication of the master device status to allow a backup device with a higher priority to assume the master device role.

A VRRP session must be configured and running.

When you change the priority of a VRRP owner, the change takes effect only for the current power cycle. The change is not saved to the startup configuration file when you save the configuration, and it is not retained across a reload or reboot. Following a reload or reboot, the VRRP owner again has priority 255.

NOTE

This task supports IPv4 VRRP only. IPv6 VRRP, VRRP-E, and IPv6 VRRP-E are not supported.

1. On the master device and from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP.

```
device(config)# router vrrp
```

3. Configure an Ethernet interface.

```
device(config)# interface ethernet 1/1/6
```

4. Enter the virtual router ID (VRID) for which the device is the VRRP owner.

```
device(config-if-e1000-1/1/6)# ip vrrp vrid 1
```

NOTE

You can assign a VRID number in the range of 1 through 255.

5. Enter a priority for this device that is lower than the priority of at least one backup device associated with the VRID.

```
device(config-if-e1000-1/1/6-vrid-1)# owner priority 99
```

6. Verify the abdication of the master device using the **show ip vrrp** command.

```
device(config-if-e1000-1/1/6-vrid-1)# show ip vrrp
```

```
Total number of VRRP routers defined: 1
Interface ethernet 1/1/6
auth-type no authentication
VRID 1
state backup
administrative-status enabled
mode owner
priority 99
current priority 99
hello-interval 1 sec
ip-address 10.53.5.1
backup routers 10.53.5.2
```

In this example, the mode shows this device as the owner of the virtual router (mode owner), but the VRRP priority for the device is only 99 and the state is now backup instead of master. The administrative status is still enabled. The output verifies that this device is now a backup device.

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.

Tracking ports and setting the VRRP priority

Configuring port tracking on an exit path interface and setting a priority on a VRRP device enables VRRP to monitor the interface. For VRRP, if the interface goes down, the device priority is set to the priority value and another backup device with a higher priority assumes the role of master. For VRRP-E, if the interface goes down, the device priority is lowered by the priority value and another backup device with a higher priority assumes the role of master.

Configure this task on the device on which the tracked interface exists.

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

```
device# configure terminal
```

2. Enter the **router vrrp** command to configure VRRP globally.

```
device(config)# router vrrp
```

3. Configure the Ethernet interface.

```
device(config)# interface ethernet 1/1/6
```

4. Enter the IP address for the interface to be used for the virtual router ID (VRID).

```
device(config-if-e1000-1/1/6)# ip address 10.53.5.3/24
```

5. Enter the following command to enter the appropriate VRRP virtual router ID (VRID) mode.

```
device(config-if-e1000-1/1/6)# ip vrrp vrid 1
```

6. Enter the **track-port** command to set the track port and priority:

```
device(config-if-e1000-1/1/6-vrid-1)# track-port ethernet 1/2/4 priority 20
```

The priority value is used when a tracked port goes down and the new priority is set to this value. Ensure that the priority value is lower than the priority set for any existing master or backup device to force a renegotiation for the master device.

The following example shows how to configure Ethernet interface 1/2/4 on virtual router 1 to be tracked; if the interface fails, the VRRP priority of the device becomes 20, forcing a negotiation for a new master device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/1/6
device(config-if-e1000-1/1/6)# ip address 10.53.5.1/24
device(config-if-e1000-1/1/6)# ip vrrp vrid 1
device(config-if-e1000-1/1/6-vrid-1)# track-port ethernet 1/2/4 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.

Disabling VRRP backup preemption

VRRP backup preemption can be disabled to avoid route flapping when a backup VRRP device that is acting as the master device could be preempted by another backup device with a higher priority value.

A VRRP or VRRP-E session must be globally enabled using the **router vrrp** or **router vrrp-extended** command in global configuration mode.

Preemption is enabled by default for VRRP and VRRP-E, but if several devices come back online with higher priorities than the original backup device, route flapping can occur as these devices preempt each other. The following steps can be used when you want to avoid a backup device acting as the master from being preempted by another backup device with a higher priority value.

1. Enter interface configuration mode.

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

2. Enter the IP address for the interface to be used for the virtual router ID (VRID).

```
device(config-if-e1000-1/1/5)# ip address 10.53.5.3/24
```

3. Enter the following command to enter the appropriate VRRP VRID mode.

```
device(config-if-e1000-1/1/5)# ip vrrp vrid 1
```

4. Enter the **non-preempt-mode** command to disable backup preemption.

```
device(config-if-e1000-1/1/5-vrid-1)# non-preempt-mode
```

Even if a backup device has a higher priority than the current backup acting as a master device, the backup device will not assume the role of the VRRP master device.

The following example disables preemption on a backup VRRP device.

```
device(config)# router vrrp
device(config)# interface ethernet 1/1/5
device(config-if-e1000-1/1/5)# ip address 10.53.5.3/24
device(config-if-e1000-1/1/5)# ip vrrp vrid 1
device(config-if-e1000-1/1/5-vrid-1)# non-preempt-mode
```

Accept mode for backup VRRP devices

Accept mode allows a backup VRRP device to respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

For each VRRP virtual routing instance, there is one master device and all other devices are backups. Accept mode allows some network management functionality for backup VRRP devices, providing the ability to respond to ping, traceroute, and Telnet packets. By default, nonowner VRRP devices do not accept packets destined for the IPv4 or IPv6 VRID addresses. Troubleshooting network connections to the VRRP nonowner master device is difficult unless accept mode is enabled.

NOTE

The accept mode functionality enables a VRRP nonowner master device to respond to ping, Telnet, and traceroute packets, but the device will not respond to SSH packets.

Enabling accept mode on a backup VRRP device

Enabling accept mode allows a backup VRRP device to respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

This task is performed on any device that is designated as a backup VRRP device, and the functionality is activated if the backup device becomes a master VRRP device. Repeat this task for all devices that are to be designated as backup devices.

NOTE

The accept mode functionality does not support SSH packets.

1. On the device designated as a backup VRRP device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP.

```
device(config)# router vrrp
```

3. Configure the Ethernet interface link.

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

Accept mode for backup VRRP devices

- Configure the IP address of the interface. All devices configured for the same virtual router ID (VRID) must be on the same subnet.

```
device(conf-if-e1000-1/1/5)# ip address 10.53.5.3/24
```

- Assign this backup device to VRID 1, the same VRID as the VRRP owner device.

```
device(conf-if-e1000-1/1/5)# ip vrrp vrid 1
```

NOTE

You can assign a VRID number in the range of 1 through 255.

- Designate this router as a backup VRRP device.

```
device(conf-if-e1000-1/1/5-vrid-1)# backup priority 110
```

While configuring a backup device, you can set a priority that is used when a master VRRP device goes offline. The backup device with the highest priority will assume the role of master device.

- Enable accept mode for this device.

```
device(conf-if-e1000-1/1/5-vrid-1)# accept-mode
```

- Exit configuration mode and return to privileged EXEC mode.

```
device(conf-if-e1000-1/1/5-vrid-1)# end
```

- Verify that accept mode is enabled.

```
device# show ip vrrp vrid 1

Interface 1/1/5
-----
auth-type no authentication
VRID 1 (index 1)
  interface 1/1/5
  state master
  administrative-status enabled
  version v2
  mode non-owner (backup)
  virtual mac aaaa.bbbb.cccc (configured)
  priority 110
  current priority 110
  track-priority 2
  hello-interval 1 sec
  accept-mode enabled
.
.
.
```

The following example enables accept mode for a backup VRRP device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/1/5
device(conf-if-e1000-1/1/5)# ip address 10.53.5.3/24
device(conf-if-e1000-1/1/5)# ip vrrp vrid 1
device(conf-if-e1000-1/1/5-vrid-1)# backup priority 110
device(conf-if-e1000-1/1/5-vrid-1)# accept-mode
```

Suppressing RIP route advertisements on VRRP backup devices

RIP route advertisement suppression can be enabled on VRRP backup devices to prevent other VRRP devices from learning multiple paths for a backed-up interface.

A VRRP or VRRP-E session with master and backup devices must be configured and running.

Normally, a VRRP or VRRP-E backup includes route information for the virtual IP address (the backed-up interface) in RIP advertisements. As a result, other devices receive multiple paths for the backed-up interface and might sometimes unsuccessfully use the path to the backup device rather than the path to the master device.

You can prevent the backups from advertising route information for the backed-up interface by enabling suppression of the advertisements.

NOTE

The command syntax is the same for VRRP and VRRP-E.

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

```
device# configure terminal
```

2. Enable RIP.

```
device(config)# router rip
```

3. Suppress RIP route advertisements.

```
device(config-rip-router)# use-vrrp-path
```

The following example suppresses RIP advertisements for the backed-up interface.

```
device# configure terminal
device(config)# router rip
device(config-rip-router)# use-vrrp-path
```

VRRP-Ev2 overview

VRRP Extended (VRRP-E) is an extended version of VRRP. VRRP-E is designed to avoid the limitations in the standards-based VRRP.

To create VRRP-E, Brocade has 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.

VRRP-E is not supported on non-Brocade devices and does not interoperate with VRRP sessions on Brocade devices.

Enabling a VRRP-E device

This task is performed on any device that is designated as a VRRP extended (VRRP-E) device. For each VRRP-E virtual routing instance, there is one master device and all other devices are backups; but, unlike VRRP, every device is configured as a backup and the device with the highest priority becomes the master VRRP-E device. Repeat this task for all devices that are to be designated as VRRP-E devices.

NOTE

Only VRRP or VRRP-E can be enabled in your network.

1. On the device designated as a VRRP-E device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP-E.

```
device(config)# router vrrp-extended
```

3. Configure the Ethernet interface link.

```
device(config-vrrpe-router)# interface ethernet 1/1/5
```

4. Configure the IP address of the interface. All devices configured for the same virtual router ID (VRID) must be on the same subnet.

```
device(config-if-e1000-1/1/5)# ip address 10.53.5.3/24
```

5. Assign the device to VRID 1.

```
device(config-if-e1000-1/1/5)# ip vrrp-extended vrid 1
```

NOTE

You can assign a VRID number in the range of 1 through 255.

6. Designate this router as a backup VRRP device.

```
device(config-if-e1000-1/1/5-vrid-1)# backup priority 110
```

While configuring a backup device, you can set a priority that is used when a master VRRP device goes offline. The backup device with the highest priority will assume the role of master device.

7. Configure the VRRP version.

```
device(config-if-e1000-1/1/5-vrid-1)# version 2
```

8. Configure the IP address of the VRID.

```
device(config-if-e1000-1/1/5-vrid-1)# ip-address 10.53.5.254
```

The IP address associated with the VRID must not be configured on any of the devices used for VRRP-E.

9. Enable the VRRP-E session.

```
device(config-if-e1000-1/1/5-vrid-1)# activate
VRRP-E router 1 for this interface is activating
```

The following example configures a VRRP-E device.

```
device# configure terminal
device(config)# router vrrp-extended
device(config-vrrpe-router)# interface ethernet 1/1/5
device(config-if-e1000-1/1/5)# ip address 10.53.5.3/24
device(config-if-e1000-1/1/5)# ip vrrp-extended vrid 1
device(config-if-e1000-1/1/5-vrid-1)# backup priority 110
device(config-if-e1000-1/1/5-vrid-1)# version 2
device(config-if-e1000-1/1/5-vrid-1)# ip-address 10.53.5.254
device(config-if-e1000-1/1/5-vrid-1)# activate
VRRP-E router 1 for this interface is activating
```

VRRP-E load-balancing using short-path forwarding

The VRRP-E Extension for Server Virtualization feature allows Brocade 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.

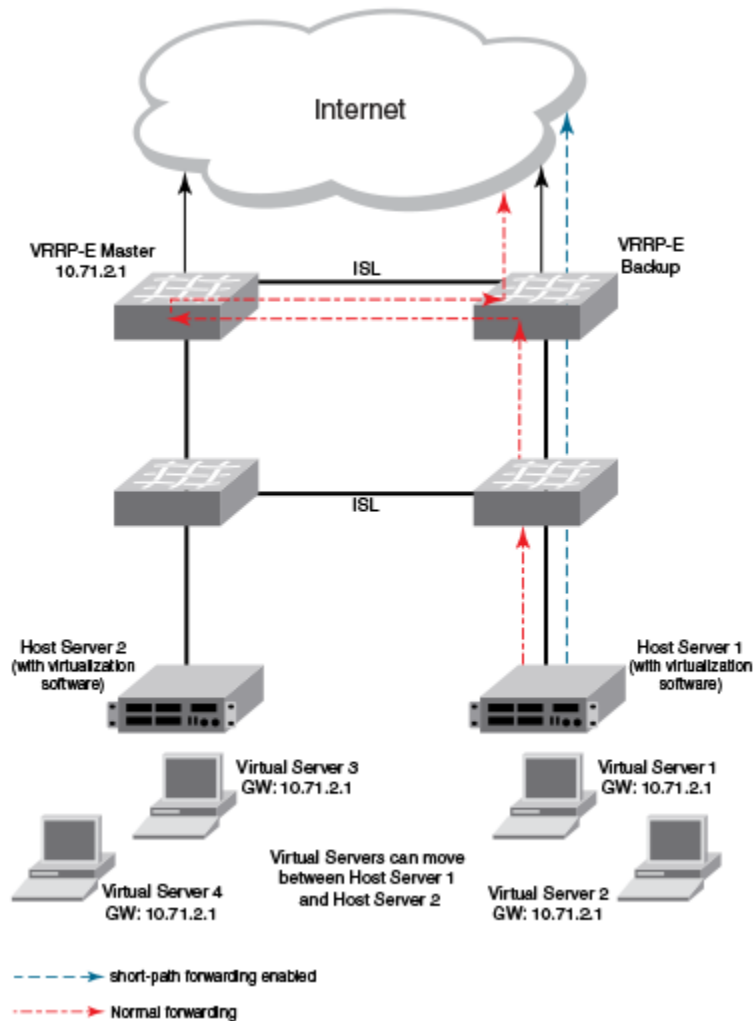
NOTE

VRRP-E load-balancing is supported only on ICX 7750 devices.

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 39 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 backup interface exists can be routed directly to the VRRP 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. Use the **configure terminal** command to enter global configuration mode.

```
device# configure terminal
```

2. To globally enable VRRP-E, enter the **router vrrp-extended** command.

```
device(config)# router vrrp-extended
```

3. Enter the **interface ve** command with an associated VLAN number.

```
device(config-vrrp-extended)# interface ve 10
```

In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned a VLAN number of 10.

4. Enter an IP address for the interface using the **ip address** command.

```
device(config-vif-10)# ip address 192.168.4.1/24
```

5. Enter the **ip vrrp-extended vrid** command with a number to assign a VRRP-E virtual router ID to the device.

```
device(config-vif-10)# ip vrrp-extended vrid 5
```

In this example, VRRP-E group configuration mode is entered.

6. Enter the **backup** command with a **priority** value to configure the device as a VRRP-E backup device.

```
device(config-vif-10-vrid-5)# backup priority 50
```

7. Enter the **ip-address** command with an IP address that is not used on any VRRP-E device interface to add a virtual IP address to the VRRP-E instance.

```
device(config-vif-10-vrid-5)# ip-address 192.168.4.254
```

8. Enter the **short-path-forwarding** command with a **revert-priority** value to configure the backup VRRP-E device as an alternate path with a specified priority.

```
device(config-vif-10-vrid-5)# short-path-forwarding revert-priority 50
```

When the backup device priority is higher than the configured **revert-priority** value, the backup router is able to perform short-path forwarding. If the backup priority is lower than the revert priority, short-path forwarding is disabled.

9. Enter the **activate** command to activate the VRRP-E instance.

```
device(config-vif-10-vrid-5)# activate
```

In the following example, short-path forwarding is configured on a backup VRRP-E device, and a revert priority threshold is configured. If the backup device priority falls below this threshold, short-path forwarding is disabled.

```
device# configure terminal
device(config)# router vrrp-extended
device(config-vrrpe-router)# interface ve 10
device(config-vif-10)# ip address 192.168.4.1/24
device(config-vif-10)# ip vrrp-extended vrid 5
device(config-vif-10-vrid-5)# backup priority 50
device(config-vif-10-vrid-5)# ip-address 192.168.4.254
device(config-vif-10-vrid-5)# short-path-forwarding revert-priority 50
device(config-vif-10-vrid-5)# activate
```

VRRP-E hitless upgrade

When the Virtual Routing Redundancy Protocol Extended (VRRP-E) hitless upgrade capability is configured, traffic loss can be avoided during the failover process when an upgrade is installed or when troubleshooting is being performed on a VRRP-E master device.

Virtual Router Redundancy Protocol Extended (VRRP-E) backup devices elect a master VRRP-E device based on the device with the highest virtual router ID (VRID) priority value. When scheduling an upgrade, the standard failover behavior involves increasing the priority value of a backup VRRP-E device to enable it to assume the role of VRRP-E master. VRRP-E hitless upgrade provides functionality to decrease the priority of the virtual router identifier (VRID) to a value of 1 for a master VRRP-E device. When the short-path forwarding functionality is also configured, the potential for traffic loss is avoided because backup devices can forward traffic. A new command-line interface (CLI) command is introduced to decrease the priority of VRIDs on the master device before an upgrade. A second use case is when troubleshooting must be performed on the master VRRP-E device.

To implement the hitless upgrade, configure short-path forwarding on all the devices assigned to the same virtual router ID (VRID) as the master device. Load the upgrade image and, before the device reloads, enable the CLI that decreases the VRID priority of the master device. If you save the configuration to the startup configuration file before the reload, the device will remain a backup device after rebooting to allow time to check that the new image is working correctly. After the hitless upgrade functionality is configured, VRRP-E activates the automatic backup by setting the master device VRID priority to one. All VRRP-E devices become backup devices and the software chooses the device with the highest VRID priority to become the master device. Using short-path forwarding where a backup device can forward traffic, any data traffic loss is avoided.

After the upgrade has been performed, the previous master device boots up as a VRRP-E backup device. To return the device to its previous role, remove the hitless upgrade configuration. Remember to save the startup configuration after the hitless upgrade command removal because the system is in maintenance mode after the VRRP-E hitless upgrade is enabled and no other device configuration changes are recommended.

VRRP-E hitless upgrade functionality is only supported in VRRP-E IPv4. This feature enables short-path forwarding support on all FastIron devices.

Configuring VRRP-E hitless upgrade

Configure the VRRP-E hitless upgrade capability to avoid traffic loss while upgrading or troubleshooting a VRRP-E master device.

Before configuring VRRP-E hitless upgrade, VRRP-E must be configured on all devices used in the VRRP-E session. To avoid any traffic loss during the failover process, enable short-path forwarding on all VRRP-E devices.

Perform this task on the master VRRP-E Layer 3 devices to configure the VRRP-E hitless upgrade capability.

1. Enter global configuration mode.

```
device# configure terminal
```

2. To globally enable VRRP-E, enter the **router vrrp-extended** command.

```
device(config)# router vrrp-extended
```

3. Configure the Virtual Ethernet (VE) interface for the VRRP-E device.

```
device(config-vrrpe-router)# interface ve 10
```

4. Configure the IP address of the interface. All devices configured for the same virtual router ID (VRID) must be on the same subnet.

```
device(config-vif-10)# ip address 192.168.4.1/24
```

5. Assign this device to VRID 5.

```
device(config-vif-10)# ip vrrp-extended vrid 5
```

6. Designate this router as a backup VRRP device. The backup device with the highest priority assumes the role of master VRRP-E device.

```
device(config-vif-10-vrid-5)# backup priority 110
```

7. Enter the **ip-address** command with an IP address that is not used on any VRRP-E device interface to add a virtual IP address to the VRRP-E instance.

```
device(config-vif-10-vrid-5)# ip-address 192.168.4.254
```

8. Enter the **short-path-forwarding** command.

```
device(config-vif-10-vrid-5)# short-path-forwarding
```

9. Activate the VRRP-E instance.

```
device(config-vif-10-vrid-5# activate
```

10. Return to virtual interface configuration mode.

```
device(config-vif-10-vrid-5# exit
```

11. Return to global configuration mode.

```
device(config-vif-10)# exit
```

12. Enter the **activate backup** command to work in conjunction with the short-path forwarding configuration to enable VRRP-E hitless upgrade.

```
device(config-vrrpe-router)# activate backup
```

In this example, VRRP-E hitless upgrade is enabled on the master VRRP-E device. The priority of the master VRRP-E device is set to 1 and the backup device with the highest priority assumes the role of the master VRRP-E device.

13. Return to global configuration mode.

```
device(config-vrrpe-router)# exit
```

14. (Optional) You can write the running configuration file to the startup configuration file to ensure that the device remains as a backup device until after the reload.

```
device(config)# write memory
```

Be aware that while the hitless upgrade is enabled, no other device configuration is recommended; the system is in maintenance mode. After your upgrade or troubleshooting is complete, remove the **activate backup** command.

In the following example, VRRP-E hitless upgrade is enabled and the running-config file is saved to the startup-config file.

```
device# configure terminal
device(config)# router vrrp-extended
device(config-vrrpe-router)# interface ve 10
device(config-vif-10)# ip address 192.168.4.1/24
device(config-vif-10)# ip vrrp-extended vrid 5
device(config-vif-10-vrid-5)# backup priority 110
device(config-vif-10-vrid-5)# ip-address 192.168.4.254
device(config-vif-10-vrid-5)# short-path-forwarding
device(config-vif-10-vrid-5)# activate
device(config-vif-10-vrid-5)# exit
device(config-vif-10)# exit
device(config-vrrpe-router)# activate backup
device(config-vrrpe-router)# exit
device(config)# write memory
```

While the VRRP-E hitless upgrade is enabled, load your upgrade image and reboot the device or perform any troubleshooting.

VRRP-E slow start timer

In a VRRP extended (VRRP-E) configuration, if a master device goes offline, the backup router with the highest priority takes over after the expiration of the dead interval timer. When the original master device is back online, you can configure a slow-start timer interval that extends the time interval beyond the dead interval before the original master device transitions back to the role of master device.

The slow-start interval allows additional time for routing protocols, for example OSPF, to converge without causing route flapping during the transition from backup device to master device. Included in the VRRP-E slow-start timer feature are track port state changes and restart options. The **use-track-port** option implements a slow-start timer for the first tracked port "up" state change, in addition to the VRRP-E initialization state. The **restart** option restarts the slow-start timer for subsequent tracked port "up" state changes.

NOTE

If you change the backup priority of a VRRP-E backup router to be higher than the priority of the original master device, the slow-start timer will not work. The original master device will take over from the backup device immediately.

Configuring a VRRP-E slow-start timer

The slow-start timer is a VRRP-E interval timer that extends beyond the dead interval during a transition from the backup device that assumed the master role to the original master device that is back online and has a higher priority.

In a VRRP extended (VRRP-E) configuration, if a master device goes offline, the backup router with the highest priority takes over after the expiration of the dead interval timer. When the original master device is back online, you can configure a slow-start timer interval that extends the time interval beyond the dead interval before the original master device transitions back to the role of master device.

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

```
device# configure terminal
```

2. To globally enable VRRP-E, enter the **router vrrp-extended** command.

```
device(config)# router vrrp-extended
```

3. Enter the **slow-start** command with options to configure the interval, in seconds, and whether tracked-port state changes trigger the slow-start interval.

```
device(config-vrrpe-router)# slow-start 40 use-track-port restart
```

In this example, the slow-start timer interval is set to 40 seconds, and the slow-start timer also runs after the first and subsequent tracked-port state changes.

```
device# configure terminal
device(config)# router vrrp-extended
device(config-vrrpe-router)# slow-start 40 use-track-port restart
```

Configuration example: ISSU upgrade using VRRP-E

Using VRRP-E, an In Service Software Upgrade (ISSU) can be performed with minimal downtime.

VRRP-E supports ISSU and combined with the short-path forwarding feature, high availability can be achieved. When a software upgrade has to be performed, the backup router can be upgraded first and after it comes back online, the VRRP-E priority can be set to be higher than the current master. A transition is initiated by the software, and with minimal packet loss, the backup router becomes the master router running the upgraded software version. Perform the following steps that utilize the configurations and network diagram.

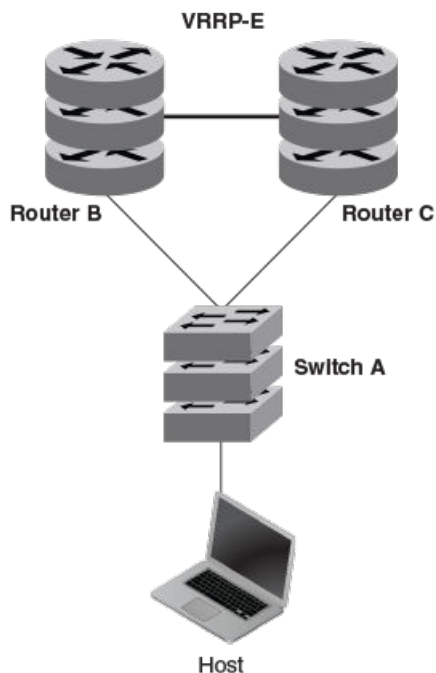
NOTE

Before configuring VRRP-E, configure your network with Layer 3 protocols using OSPF and RIP.

1. On Router B and Router C in the diagram apply the example configurations.
2. The software selects Router C as the master VRRP-E device because the priority and IP address are higher than Router B.
3. Upgrade the software version on Router B, the backup router, and reload.
4. Router B comes online and joins the network.
5. Increase the priority on Router B using the **backup priority 254** command and options in VRID interface configuration mode.
6. The software transitions the role of VRRP-E master to Router B with only 30 milliseconds of packet loss.
7. Upgrade the software version on Router C, which has become the backup router, and reload.
8. Router C comes online and joins the network.

Configuration example: ISSU upgrade using VRRP-E

FIGURE 40 VRRP-E network setup for ISSU



Router B configuration

The following example configuration configures VRRP-E using the short-path forwarding feature. On this device, the priority value for VRID 23 is set to 50.

```

configure terminal
router vrrp-extended
interface ve 123
ip address 192.168.4.11 255.255.255.0
ip vrrp-extended vrid 23
backup priority 50
advertise backup
ip-address 192.168.4.254
short-path-forwarding
activate

```

Router C configuration

The following example configuration configures VRRP-E using the short-path forwarding feature. On this device, the priority value for VRID 23 is set to 250.

```

configure terminal
router vrrp-extended
interface ve 123
ip address 192.168.4.12 255.255.255.0
ip vrrp-extended vrid 23
backup priority 250
advertise backup
ip-address 192.168.4.254
short-path-forwarding
activate

```

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 ip vrrp** command with the **vrid** option and a virtual router ID (VRID) to display IPv4 VRRP configuration information about VRID 1.

```
device# show ip vrrp vrid 1

Interface 1/1/1
-----
auth-type no authentication
VRID 1 (index 1)
interface 1/1/1
state master
administrative-status enabled
version v2
mode owner
virtual mac aaaa.bbbb.cccc (configured)
priority 255
current priority 255
track-priority 2
hello-interval 1 sec
backup hello-interval 6
```

2. Enter the **show ip vrrp brief** command.

```
device(config)# show ip vrrp brief

Total number of VRRP routers defined: 2
Flags Codes - P:Preempt 2:V2 3:V3 S:Short-Path-Fwd
Inte- VRID  Current  Flags    State   Master IP Backup IP  Virtual IP
rface   Priority                                     Address  Address  Address
-----
1/1/1 10    255      P2-     Master  Local   Unknown  10.30.30.2
1/1/3 13     100      P2-     Master  Local   Unknown  10.13.13.3
```

This example displays summary information about VRRP sessions.

3. Enter the **show ip vrrp-extended statistics** command for Ethernet interface 1/1/5.

```
device# show ip vrrp-extended statistics ethernet 1/1/5

Interface 1/1/5
-----
VRID 2
- number of transitions to backup state = 1
- number of transitions to master state = 1
- total number of vrrp-extended packets received = 0
  . received backup advertisements = 0
  . received packets with zero priority = 0
  . received packets with invalid type = 0
  . received packets with invalid authentication type = 0
  . received packets with authentication type mismatch = 0
  . received packets with authentication failures = 0
  . received packets dropped by owner = 0
  . received packets with ttl errors = 0
  . received packets with ipv6 address mismatch = 0
  . received packets with advertisement interval mismatch = 0
  . received packets with invalid length = 0
- total number of vrrp-extended packets sent = 2004
  . sent backup advertisements = 0
  . sent packets with zero priority = 0
- received neighbor solicitation packets dropped = 0
- received proxy neighbor solicitation packets dropped = 0
- received ip packets dropped = 0
```

Clearing VRRPv2 statistics

VRRPv2 session counters can be cleared using a CLI command.

Ensure that VRRPv2 or VRRP-Ev2 is configured and enabled in your network.

To determine the effect of clearing the VRRP statistics, an appropriate **show** command is entered before and after the **clear** command.

1. Enter the **exit** command to return to privileged EXEC mode.
2. Enter the **show ip vrrp statistics** command for Ethernet interface 1/1/5.

```
device# show ip vrrp statistics ethernet 1/1/5

Interface 1/1/5
-----
VRID 2
- number of transitions to backup state = 1
- number of transitions to master state = 1
- total number of vrrp packets received = 0
  . received backup advertisements = 0
  . received packets with zero priority = 0
  .
  .
- total number of vrrp packets sent = 2004
  . sent backup advertisements = 6
  . sent packets with zero priority = 0
- received neighbor solicitation packets dropped = 0
```

3. Enter the **clear ip vrrp statistics** command.

```
device# clear ip vrrp statistics
```


4. Enter the **show ip vrrp statistics** command for Ethernet interface 1/1/5.

```
device# show ip vrrp statistics ethernet 1/1/5

Interface 1/1/5
-----
VRID 2
- number of transitions to backup state = 0
- number of transitions to master state = 0
- total number of vrrp packets received = 0
  . received backup advertisements = 0
  . received packets with zero priority = 0
.
.
.
- total number of vrrp packets sent = 8
  . sent backup advertisements = 0
  . sent packets with zero priority = 0
- received neighbor solicitation packets dropped = 0
```

In this show output for a specified interface after the **clear ip vrrp statistics** command has been entered, you can see that the statistical counters have been reset. Although some of the counters are showing numbers because VRRP traffic is still flowing, the numbers are much lower than in the initial **show ip vrrp statistics** command output.

VRRPv3

• VRRPv3 overview.....	483
• Enabling an IPv6 VRRPv3 owner device.....	484
• Enabling an IPv6 VRRPv3 backup device.....	485
• Enabling an IPv4 VRRPv3 owner device.....	486
• Enabling an IPv4 VRRPv3 backup device.....	487
• Tracked ports and track priority with VRRP and VRRP-E.....	488
• Accept mode for backup VRRP devices.....	489
• Alternate VRRPv2 checksum for VRRPv3 IPv4 sessions.....	491
• Automatic generation of a virtual link-local address for VRRPv3.....	493
• Displaying VRRPv3 statistics.....	495
• Clearing VRRPv3 statistics.....	496
• VRRP-Ev3 Overview.....	496
• Enabling an IPv6 VRRP-Ev3 device.....	497
• Displaying and clearing VRRP-Ev3 statistics.....	498

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. Brocade has implemented IPv6 VRRPv3 functionality to comply with RFC 5798 and will interoperate comfortably with other vendors that support RFC 5798.

Enabling an IPv6 VRRPv3 owner device

This task is performed on the device that is designated as the owner VRRP device because the IPv6 address of one of its physical interfaces is assigned as the IP address of the virtual router. For each VRRP session, there are master and backup routers, and the owner router is elected, by default, as the master router.

NOTE

When implementing IPv6 VRRPv3 across a network with devices from other vendors, be aware of a potential interoperability issue. Brocade has implemented IPv6 VRRPv3 functionality to comply with RFC 5798 and will interoperate well with other vendors that support RFC 5798.

1. On the device designated as the owner VRRP device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Before enabling IPv6 VRRP, you must globally enable IPv6 routing.

```
device(config)# ipv6 unicast-routing
```

3. Globally enable IPv6 VRRP.

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

4. Configure the Ethernet interface link for the owner device.

```
device(config-ipv6-vrrp-router)# interface ethernet 1/1/5
```

5. Configure the IPv6 address of the interface.

```
device(config-if-e1000-1/1/5)# ipv6 address fd2b::2/64
```

6. Assign the owner device to the virtual router ID (VRID) 2.

```
device(config-if-e1000-1/1/5)# ipv6 vrrp vrid 2
```

NOTE

You can assign a VRID number in the range of 1 through 255.

7. Designate this router as the VRRP owner device.

```
device(config-if-e1000-1/1/5-vrid-2)# owner
```

8. Configure the VRRP version.

```
device(config-if-e1000-1/1/5-vrid-2)# version 3
```

9. Assign an IPv6 link-local address to the VRID for use in the local network.

```
device(config-if-e1000-1/1/5-vrid-2)# ipv6-address fe80::768e:f8ff:fe2a:0099
```

10. Assign a global IPv6 address to the VRID.

```
device(config-if-e1000-1/1/5-vrid-2)# ipv6-address fd2b::2
```

11. Enable the VRRP session.

```
device(config-if-e1000-1/1/5-vrid-2)# activate
```

The following example configures a VRRP owner device.

```
device# configure terminal
device(config)# ipv6 unicast-routing
device(config)# ipv6 router vrrp
device(config-ipv6-vrrp-router)# interface ethernet 1/1/5
device(config-if-e1000-1/1/5)# ipv6 address fd2b::2/64
device(config-if-e1000-1/1/5)# ipv6 vrrp vrid 2
device(config-if-e1000-1/1/5-vrid-2)# owner
device(config-if-e1000-1/1/5-vrid-2)# version 3
device(config-if-e1000-1/1/5-vrid-2)# ipv6-address fe80::768e:f8ff:fe2a:0099
device(config-if-e1000-1/1/5-vrid-2)# ipv6-address fd2b::2
device(config-if-e1000-1/1/5-vrid-2)# activate
```

Enabling an IPv6 VRRPv3 backup device

This task is performed on all devices that are designated as backup VRRPv3 devices. Initially a backup priority is set to 100. For each VRRPv3 session, there are master and backup routers, and the IPv6 address assigned here to the VRID is the IPv6 address of the master router. The task is repeated on each backup VRRPv3 device with corresponding changes to the interface number and IPv6 address of the interface.

NOTE

When implementing IPv6 VRRPv3 across a network with devices from other vendors, be aware of a potential interoperability issue. Brocade has implemented IPv6 VRRPv3 functionality to comply with RFC 5798 and will interoperate well with other vendors that support RFC 5798.

1. On the device designated as a backup VRRPv3 device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable IPv6 VRRP.

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

3. Configure the Ethernet interface link for the owner device.

```
device(config-ipv6-vrrp-router)# interface ethernet 1/1/4
```

4. Configure the IPv6 address of the interface.

```
device(config-if-e1000-1/1/4)# ipv6 address fd2b::3/64
```

5. Assign the backup device to the virtual router ID (VRID) 2.

```
device(config-if-e1000-1/1/4)# ipv6 vrrp vrid 2
```

NOTE

You can assign a VRID number in the range of 1 through 255.

6. Designate this router as a VRRPv3 backup device and assign it a priority of 100.

```
device(config-if-e1000-1/1/4-vrid-2)# backup priority 100
```

7. Configure the VRRP version.

```
device(config-if-e1000-1/1/4-vrid-2)# version 3
```

Enabling an IPv4 VRRPv3 owner device

- By default, backup VRRP devices do not send hello messages to advertise themselves to the master. Use the following command to enable a backup router to send hello messages to the master VRRP device.

```
device(config-if-e1000-1/1/4-vrid-2) # advertise backup
```

- Assign the IPv6 link-local address to the VRID for use in the local network.

```
device(config-if-e1000-1/1/4-vrid-2) # ipv6-address fe80::768e:f8ff:fe2a:0099
```

- Assign the global IPv6 address to the VRID.

```
device(config-if-e1000-1/1/4-vrid-2) # ipv6-address fd2b::2
```

- Enable the VRRP session.

```
device(config-if-e1000-1/1/4-vrid-2) # activate
```

The following example configures an IPv6 VRRPv3 backup device.

```
device# configure terminal
device(config)# ipv6 router vrrp
device(config-ipv6-vrrp-router)# interface ethernet 1/1/4
device(config-if-e1000-1/1/4)# ipv6 address fd2b::3/64
device(config-if-e1000-1/1/4)# ipv6 vrrp vrid 2
device(config-if-e1000-1/1/4-vrid-2) # backup priority 100
device(config-if-e1000-1/1/4-vrid-2) # version 3
device(config-if-e1000-1/1/4-vrid-2) # advertise backup
device(config-if-e1000-1/1/4-vrid-2) # ipv6-address fe80::768e:f8ff:fe2a:0099
device(config-if-e1000-1/1/4-vrid-2) # ipv6-address fd2b::2
device(config-if-e1000-1/1/4-vrid-2) # activate
```

Enabling an IPv4 VRRPv3 owner device

VRRPv3 supports IPv4 sessions as well as IPv6 sessions. To configure a VRRPv3 session for IPv4, assign a virtual router group with the VRRP version set to 3. This task is performed on the device that is designated as the owner VRRP device because the IP address of one of its physical interfaces is assigned as the IP address of the virtual router.

- On the device designated as the owner VRRP device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

- Globally enable VRRP.

```
device(config)# router vrrp
```

- Configure an Ethernet interface.

```
device(config)# interface ethernet 1/1/6
```

- Configure the IP address of the interface.

```
device(config-if-e1000-1/1/6) # ip address 10.53.5.1/24
```

- Assign the virtual router ID (VRID) 1 to the interface.

```
device(config-if-e1000-1/1/6) # ip vrrp vrid 1
```

NOTE

You can assign a VRID number in the range of 1 through 255.

- Designate this router as the VRRP owner device.

```
device(config-if-e1000-1/1/6-vrid-1)# owner
```

- Configure the VRRP version.

```
device(config-if-e1000-1/1/6-vrid-1)# version 3
```

In this step, VRRPv3 is selected.

- Configure the IP address of the VRID.

```
device(config-if-e1000-1/1/6-vrid-1)# ip-address 10.53.5.1
```

- Enable the VRRP session.

```
device(config-if-e1000-1/1/6-vrid-1)# activate
```

The following example configures an IPv4 VRRPv3 owner device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/1/6
device(config-if-e1000-1/1/6)# ip address 10.53.5.1/24
device(config-if-e1000-1/1/6)# ip vrrp vrid 1
device(config-if-e1000-1/1/6-vrid-1)# owner
device(config-if-e1000-1/1/6-vrid-1)# version 3
device(config-if-e1000-1/1/6-vrid-1)# ip-address 10.53.5.1
device(config-if-e1000-1/1/6-vrid-1)# activate
VRRP router 1 for this interface is activating
```

Enabling an IPv4 VRRPv3 backup device

VRRPv3 supports IPv4 sessions as well as IPv6 sessions. To configure a VRRPv3 session for IPv4, assign a virtual router group with the VRRP version set to 3. This task is performed on any device that is designated as an IPv4 backup VRRPv3 device. For each VRRP virtual routing instance, there is one master device and all other devices are backups. Repeat this task on all devices that are to be designated as backup devices.

- On a device designated as a backup VRRP device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

- Globally enable VRRP.

```
device(config)# router vrrp
```

- Configure the Ethernet interface.

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

- Configure the IP address of the interface. All devices configured for the same virtual router ID (VRID) must be on the same subnet.

```
device(config-if-e1000-1/1/5)# ip address 10.53.5.3/24
```

Tracked ports and track priority with VRRP and VRRP-E

- Assign the same VRID as the VRID used by the owner device.

```
device(config-if-e1000-1/1/5)# ip vrrp vrid 1
```

NOTE

You can assign a VRID number in the range of 1 through 255.

- Designate this router as a backup VRRP device.

```
device(config-if-e1000-1/1/5-vrid-1)# backup priority 110
```

While configuring a backup device, you can set a priority that is used when a master VRRP device goes offline. The backup device with the highest priority will assume the role of master device.

- Set the VRRP version to 3 to indicate that this is VRRPv3 session for IPv4.

```
device(config-if-e1000-1/1/5-vrid-1)# version 3
```

- Configure the IP address of the VRID.

```
device(config-if-e1000-1/1/5-vrid-1)# ip-address 10.53.5.1
```

The VRID IP address is the same virtual IP address that you used for the VRRP owner device.

- Enable the VRRP session.

```
device(config-if-e1000-1/1/5-vrid-1)# activate
VRRP router 1 for this interface is activating
```

The following example configures a VRRP owner device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/1/5
device(config-if-e1000-1/1/5)# ip address 10.53.5.3/24
device(config-if-e1000-1/1/5)# ip vrrp vrid 1
device(config-if-e1000-1/1/5-vrid-1)# backup priority 110
device(config-if-e1000-1/1/5-vrid-1)# version 3
device(config-if-e1000-1/1/5-vrid-1)# ip-address 10.53.5.1
device(config-if-e1000-1/1/5-vrid-1)# activate
VRRP router 1 for this interface is activating
```

Tracked ports and track priority with VRRP and VRRP-E

Port tracking allows interfaces not configured for VRRP or VRRP-E to be monitored for link-state changes that can result in dynamic changes to the VRRP device priority.

A tracked port allows you to monitor the state of the interfaces on the other end of a route path. A tracked interface also allows the virtual router to lower its priority if the exit path interface goes down, allowing another virtual router in the same VRRP (or VRRP-E) group to take over. When a tracked interface returns to an up state, the configured track priority is added to the current virtual router priority value.

The following conditions and limitations exist for tracked ports:

- Track priorities must be lower than VRRP or VRRP-E priorities.
- The dynamic change of router priority can trigger a master device switchover if preemption is enabled. However, if the router is an owner, the master device switchover will not occur.
- The maximum number of interfaces that can be tracked for a virtual router is 16.
- Port tracking is allowed for physical interfaces and port channels.

Tracking ports and setting VRRP priority using VRRPv3

Configuring port tracking on an exit path interface and setting a priority on a VRRPv3 device enables VRRPv3 to monitor the interface. For VRRPv3, if the interface goes down, the device priority is set to the priority value and another backup device with a higher priority assumes the role of master. For VRRP-Ev3, if the interface goes down, the device priority is lowered by the priority value and another backup device with a higher priority assumes the role of master.

Before enabling IPv6 VRRPv3, you must globally enable IPv6 routing using the **ipv6 unicast-routing** command.

Configure this task on the device on which the tracked interface exists.

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

```
device# configure terminal
```

2. Enter the **ipv6 router vrrp** command to configure VRRPv3 globally.

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

3. Configure the Ethernet interface.

```
device(config)# interface ethernet 1/1/6
```

4. Enter the IPv6 address for the interface to be used for the virtual router ID (VRID).

```
device(config-if-e1000-1/1/6)# ipv6 address fd2b::2/64
```

5. Enter the following command to enter the appropriate VRRPv3 virtual router ID (VRID) mode.

```
device(config-if-e1000-1/1/6)# ipv6 vrrp vrid 1
```

6. Enter the **track-port** command to set the tracked port and priority:

```
device(config-if-e1000-1/1/6-vrid-1)# track-port ethernet 1/2/4 priority 20
```

The priority value is used when a tracked port goes down and the new priority is set to this value. Ensure that the priority value is lower than the priority set for any existing master or backup device to force a renegotiation for the master device.

The following example shows how to configure interface Ethernet 1/2/4 on virtual router 1 to be tracked; if the interface fails, the IPv6 VRRPv3 priority of the device becomes 20, forcing a negotiation for a new master device.

```
device# configure terminal
device(config)# ipv6 router vrrp
device(config)# interface ethernet 1/1/6
device(config-if-e1000-1/1/6)# ipv6 address fd2b::2/64
device(config-if-e1000-1/1/6)# ipv6 vrrp vrid 1
device(config-if-e1000-1/1/6-vrid-1)# track-port ethernet 1/2/4 priority 20
```

Accept mode for backup VRRP devices

Accept mode allows a backup VRRP device to respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

For each VRRP virtual routing instance, there is one master device and all other devices are backups. Accept mode allows some network management functionality for backup VRRP devices, providing the ability to respond to ping, traceroute, and Telnet packets. By default, nonowner VRRP devices do not accept packets destined for the IPv4 or IPv6 VRID addresses. Troubleshooting network connections to the VRRP nonowner master device is difficult unless accept mode is enabled.

NOTE

The accept mode functionality enables a VRRP nonowner master device to respond to ping, Telnet, and traceroute packets, but the device will not respond to SSH packets.

Enabling accept mode on a backup VRRP device

Enabling accept mode allows a backup VRRP device to respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

This task is performed on any device that is designated as a backup VRRP device, and the functionality is activated if the backup device becomes a master VRRP device. Repeat this task for all devices that are to be designated as backup devices.

NOTE

The accept mode functionality does not support SSH packets.

1. On the device designated as a backup VRRP device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable VRRP.

```
device(config)# router vrrp
```

3. Configure the Ethernet interface link.

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

4. Configure the IP address of the interface. All devices configured for the same virtual router ID (VRID) must be on the same subnet.

```
device(conf-if-e1000-1/1/5)# ip address 10.53.5.3/24
```

5. Assign this backup device to VRID 1, the same VRID as the VRRP owner device.

```
device(conf-if-e1000-1/1/5)# ip vrrp vrid 1
```

NOTE

You can assign a VRID number in the range of 1 through 255.

6. Designate this router as a backup VRRP device.

```
device(conf-if-e1000-1/1/5-vrid-1)# backup priority 110
```

While configuring a backup device, you can set a priority that is used when a master VRRP device goes offline. The backup device with the highest priority will assume the role of master device.

7. Enable accept mode for this device.

```
device(conf-if-e1000-1/1/5-vrid-1)# accept-mode
```

8. Exit configuration mode and return to privileged EXEC mode.

```
device(conf-if-e1000-1/1/5-vrid-1)# end
```

9. Verify that accept mode is enabled.

```
device# show ip vrrp vrid 1

Interface 1/1/5
-----
auth-type no authentication
VRID 1 (index 1)
  interface 1/1/5
    state master
    administrative-status enabled
    version v2
    mode non-owner (backup)
    virtual mac aaaa.bbbb.cccc (configured)
    priority 110
    current priority 110
    track-priority 2
    hello-interval 1 sec
    accept-mode enabled
  .
  .
  .
```

The following example enables accept mode for a backup VRRP device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/1/5
device(conf-if-e1000-1/1/5)# ip address 10.53.5.3/24
device(conf-if-e1000-1/1/5)# ip vrrp vrid 1
device(conf-if-e1000-1/1/5-vrid-1)# backup priority 110
device(conf-if-e1000-1/1/5-vrid-1)# accept-mode
```

Alternate VRRPv2 checksum for VRRPv3 IPv4 sessions

If VRRPv3 is configured on a Brocade 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 Brocade 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 Brocade 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 VRRPv2 checksum computation method in a VRRPv3 IPv4 session

An alternate VRRPv2-style checksum can be configured in a VRRPv3 IPv4 session for compatibility with third-party network devices.

VRRPv3 uses the v3 checksum computation method by default for both IPv4 and IPv6 sessions on Brocade devices. Third-party devices may have only a VRRPv2-style checksum computation available for a VRRPv3 IPv4 session. The **use-v2-checksum** command is entered in interface configuration mode.

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

```
device# configure terminal
```

2. Enable VRRP globally.

```
device(config)# router vrrp
```

3. Enter the **interface** command with an interface type and number.

```
device(config)# interface ethernet 1/2/4
```

4. To configure a VRRP virtual routing ID, use the **ip vrrp vrid** command with an associated ID number.

```
device(config-if-e1000-1/2/4)# ip vrrp vrid 14
```

5. To enable VRRP version 3 (VRRPv3), enter the **version** command with a version number of v3.

```
device(config-if-e1000-1/2/4-vrid-14)# version v3
```

6. To enable the v2 checksum computation method in an IPv4 VRRPv3 session, use the **use-v2-checksum** command in VRRP configuration mode.

```
device(config-if-e1000-1/2/4-vrid-14)# use-v2-checksum
```

7. Enter the IP address for the interface using the **ip-address** command.

```
device(config-if-e1000-1/2/4-vrid-14)# ip-address 10.14.14.99
```

8. To activate the interface, enter the **activate** command.

```
device(config-if-e1000-1/2/4-vrid-14)# activate
```

The following example shows the v2 checksum computation method enabled for an VRRPv3 IPv4 session on a Brocade device.

```
device# configure terminal
device(config)# router vrrp
device(config)# interface ethernet 1/2/4
device(config-if-e1000-1/2/4)# ip vrrp vrid 14
device(config-if-e1000-1/2/4-vrid-14)# version v3
device(config-if-e1000-1/2/4-vrid-14)# use-v2-checksum
device(config-if-e1000-1/2/4-vrid-14)# ip-address 10.14.14.99
device(config-if-e1000-1/2/4-vrid-14)# activate
```

Displaying alternate VRRPv2 checksum settings

The verification of the use of the alternate VRRPv2-style checksum for VRRPv3 IPv4 sessions is achieved using several CLI commands.

The following steps are both optional and can be used to verify that the alternate VRRPv2-style checksum computation command, **use-v2-checksum**, has been set for VRRPv3 IPv4 sessions.

1. Use the **show running-config** command to verify that the **use-v2-checksum** command has been configured for a specified interface. Only part of the output is displayed.

```
device# show running-config

interface ethernet 1/2/4
ip address 10.14.14.2/24
ip vrrp vrid 14
  backup
  ip-address 10.14.14.99
  use-v2-checksum
exit
```

2. Use the **show ip vrrp** command with a virtual router ID number to display the current settings of a specific VRRP session, including the **use-v2-checksum** command, if configured.

```
device# show ip vrrp vrid 14

Interface 1/2/4
-----
auth-type no authentication

VRID 14 (index 1)
  interface 1/2/4
  state initialize
  administrative-status disabled
  version v3 - use-v2-checksum
  mode non-owner(backup)
  virtual mac 0000.5e00.010e
  priority 100
  current priority 100
  track-priority 1
  hello-interval 1 sec
  backup hello-interval 60 sec
  slow-start timer (configured) 0 sec
  advertise backup disabled
  dead-interval 3500 ms
  preempt-mode true
  ip-address 10.14.14.99
```

Automatic generation of a virtual link-local address for VRRPv3

The virtual MAC address is used to automatically generate the IPv6 virtual link-local address to simplify the configuration of IPv6 VRRP and standardize implementations across vendor platforms. Subsequent VRRPv3 advertisements carry the auto-generated virtual link-local address.

The default VRRPv3 implementation allows only the link-local address that is configured on a physical interface to be used as the virtual IPv6 address of a VRRPv3 session. This limits configuring a link-local address for each VRRP instance on the same physical interface because there can be only one link-local address per physical interface.

When IPv6 link-local address auto-generation is configured for IPv6 VRRP, a virtual IPv6 link-local address is generated automatically using the EUI-64 result of the virtual MAC address. The virtual IPv6 link-local address is generated for a specific VRRP instance and the virtual link-local address is carried in VRRPv3 advertisements. The auto-generation process is defined in RFC 5798 allowing cross-vendor platform support. This ability to generate a link-local address automatically depends on the existence of a consistent virtual MAC address in the local network.

If the virtual link-local address is configured manually, the configured address takes precedence over the automatically generated address. The administrator should ensure that the configured virtual link-local address is consistent across all routers in the LAN. When the manually configured address is removed, the auto-generated address is used.

If there is a mismatch in the IPv6 addresses field, Brocade devices drop the advertisements that are sent by backup VRRP routers. The advertisements from the master VRRP router are not dropped regardless of the IPv6 address comparison. The virtual MAC must be consistent on the local network. When the virtual MAC is modified, the virtual link-local address is regenerated.

As a Brocade proprietary protocol, VRRP Extended version 3 (VRRP-Ev3) is not supported.

Automatic generation of a virtual link-local address for VRRPv3

Assigning an auto-generated link-local IPv6 address for a VRRPv3 cluster

A virtual link-local IPv6 address can be auto-generated and assigned as the virtual IPv6 address of a VRRPv3 session.

The default VRRPv3 implementation allows only the link-local address that is configured on a physical interface to be used as the virtual IPv6 address of a VRRPv3 session. This limits configuring a link-local address for each VRRP instance on the same physical interface because there can be only one link-local address per physical interface. To auto-generate and assign a virtual link-local IPv6 address as the virtual IPv6 address of a VRRPv3 cluster, use the following steps on either an IPv6 VRRPv3 owner or backup device.

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

```
device# configure terminal
```

2. Globally enable IPv6 VRRP.

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

3. Configure the Ethernet interface link for the owner device.

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

4. Configure the global IPv6 address of the interface.

```
device(config-vif-3)# ipv6 address fd3b::3/64
```

5. Assign the device to virtual router ID (VRID) 2.

```
device(config-vif-3)# ipv6 vrrp vrid 2
```

NOTE

You can assign a VRID number in the range of 1 through 255.

6. Designate this router as a VRRPv3 owner device.

```
device(config-vif-3-vrid-2)# owner
```

7. Automatically generate the IPv6 link-local address for the VRID for use in the local network.

```
device(config-vif-3-vrid-2)# ipv6-address auto-gen-link-local
```

8. Enable the VRRP session.

```
device(config-vif-3-vrid-2)# activate
```

The following example shows the auto-generation of a virtual link-local IPv6 address and its allocation as the virtual IPv6 address of a VRRPv3 session on an IPv6 VRRPv3 owner router.

```
device# configure terminal
device(config)# ipv6 router vrrp
device(config)# interface ve 3
device(config-vif-3)# ipv6 address fd3b::3/64
device(config-vif-3)# ipv6 vrrp vrid 2
device(config-vif-3-vrid-2)# owner
device(config-vif-3-vrid-2)# ipv6-address auto-gen-link-local
device(config-vif-3-vrid-2)# activate
```

Displaying VRRPv3 statistics

Various show commands can display statistical information about IPv6 VRRP configurations.

Before displaying statistics, VRRPv3 must be configured and enabled in your network to generate traffic.

Use one or more of the following commands to display VRRPv3 information. The commands do not have to be entered in this order.

1. Use the **exit** command to return to privileged EXEC mode, if required.
2. Enter the **show ipv6 vrrp** command to display IPv6 VRRPv3 configuration information.

```
device(config)# show ipv6 vrrp

Total number of VRRP routers defined: 1
Interface 1/1/3
-----
auth-type no authentication
VRID 13 (index 2)
interface 1/1/3
state master
administrative-status enabled
version v3
mode non-owner(backup)
virtual mac 0000.5e00.0217
priority 100
current priority 100
track-priority 1
hello-interval 1000 ms
backup hello-interval 60000 ms
advertise backup disabled
dead-interval 3000 ms
preempt-mode true
ipv6-address fd2b::1
next hello sent in 700 ms
short-path-forwarding disabled
```

3. To view detailed statistical information about IPv6 VRRPv3, enter the **show ipv6 vrrp statistics** command.

```
device# show ipv6 vrrp statistics

Global IPv6 VRRP statistics
-----
- received vrrp packets with checksum errors = 0
- received vrrp packets with invalid version number = 0
- received vrrp packets with unknown or inactive vrid = 0
Interface 1/1/3
-----
VRID 13
- number of transitions to backup state = 1
- number of transitions to master state = 1
- total number of vrrp packets received = 0
. received backup advertisements = 19
. received packets with zero priority = 0
. received packets with invalid type = 0
. received packets with invalid authentication type = 0
. received packets with authentication type mismatch = 0
. received packets with authentication failures = 0
. received packets dropped by owner = 0
. received packets with ttl errors = 0
. received packets with ipv6 address mismatch = 0
. received packets with advertisement interval mismatch = 0
. received packets with invalid length = 0
- total number of vrrp packets sent = 1175
. sent backup advertisements = 0
. sent packets with zero priority = 0
- received neighbor solicitation packets dropped = 0
- received proxy neighbor solicitation packets dropped = 0
- received ipv6 packets dropped = 0
```

Clearing VRRPv3 statistics

VRRPv3 session counters can be cleared 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 Brocade 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, Brocade 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.

VRRP-Ev3 is not supported on non-Brocade devices and does not interoperate with VRRPv2 or VRRPv3 sessions on Brocade devices.

Enabling an IPv6 VRRP-Ev3 device

This task is performed on any device that is designated as a VRRP extended version 3 (VRRP-Ev3) device. For each VRRP-Ev3 virtual routing instance, there is one master device and all other devices are backups; but, unlike VRRPv3, every device is configured as a backup and the device with the highest priority becomes the master device. Repeat this task for all devices that are to be designated as VRRP-Ev3 devices.

NOTE

Only VRRPv3 or VRRP-Ev3 can be enabled in your network.

- On the device designated as a VRRP-Ev3 device, from privileged EXEC mode, enter global configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

- Globally enable VRRP-Ev3.

```
device(config)# ipv6 router vrrp-extended
```

- Configure the Ethernet interface link.

```
device(config-ipv6-vrrpe-router)# interface ethernet 1/1/7
```

- Configure the IPv6 address of the interface. All devices configured for the same virtual router ID (VRID) must be on the same subnet.

```
device(config-if-e1000-1/1/7)# ipv6 address fd4b::4/64
```

- Assign the device to VRID 4.

```
device(config-if-e1000-1/1/7)# ipv6 vrrp-extended vrid 4
```

NOTE

You can assign a VRID number in the range of 1 through 255.

- Designate this router as a backup VRRPv3 device. All VRRP-Ev3 devices are initially configured as backup devices; the device with the highest priority assumes the role of master device.

```
device(config-if-e1000-1/1/7-vrid-4)# backup priority 110
```

While configuring a backup device, you can set a priority that is used when the designated master VRRP device goes offline. The backup device with the highest priority will assume the role of master device.

- Configure the VRRP version.

```
device(config-if-e1000-1/1/7-vrid-4)# version 3
```

- Configure an IPv6 link-local address for the VRID.

```
device(config-if-e1000-1/1/7-vrid-4)# ipv6-address fe80::768e:f8ff:fe2a:0089
```

Displaying and clearing VRRP-Ev3 statistics

- Configure a global IPv6 address for the VRID.

```
device(config-if-e1000-1/1/7-vrid-4)# ipv6-address fd4b::99
```

The IPv6 address associated with the VRID must not be configured on any of the devices used for VRRP-Ev3.

- Enable the VRRP session.

```
device(config-if-e1000-1/1/7-vrid-4)# activate
VRRP-E router 4 for this interface is activating
```

The following example configures a backup VRRP-Ev3 device.

```
device# configure terminal
device(config)# ipv6 router vrrp-extended
device(config-ipv6-vrrpe-router)# interface ethernet 1/1/7
device(config-if-e1000-1/1/7)# ipv6 address fd4b::4/64
device(config-if-e1000-1/1/7)# ipv6 vrrp-extended vrid 4
device(config-if-e1000-1/1/7-vrid-4)# backup priority 50
device(config-if-e1000-1/1/7-vrid-4)# version 3
device(config-if-e1000-1/1/7-vrid-4)# ipv6-address fe80::768e:f8ff:fe2a:0089
device(config-if-e1000-1/1/7-vrid-4)# ipv6-address fd4b::99
device(config-if-e1000-1/1/7-vrid-4)# activate
VRRP-E router 4 for this interface is activating
```

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.

- Use the **exit** command to return to privileged EXEC mode, if required.
- Enter the **show ipv6 vrrp-extended brief** command to display VRRP-Ev3 summary information.

```
device(config)# show ipv6 vrrp-extended brief

Total number of VRRP routers defined: 1
Flags Codes - P:Preempt 2:V2 3:V3 S:Short-Path-Fwd
Intf      VRID CurrPrio Flags State  Master-IPv6 Backup-IPv6 Virtual-IPv6
Address                                     Address      Address
-----
1/1/3     2      100      P3-   Master Local      fd2b::2      fd2b::99
```

3. Enter the **show ipv6 vrrp-extended vrid 1** command to display detailed IPv6 VRRP-E configuration information about VRID 1.

```
device# show ipv6 vrrp-extended vrid 1
Interface 1/1/1
-----
auth-type md5-authentication
VRID 1 (index 1)
interface 1/1/1
state master
administrative-status enabled
mode non-owner(backup)
virtual mac dddd.eeee.ffff (configured)
priority 100
current priority 100
track-priority 5
hello-interval 1 sec
backup hello-interval 60 sec
advertise backup disabled
dead-interval 0 ms
preempt-mode true
virtual ipv6 address 10:20:1::100
```

4. Enter the **clear ipv6 vrrp-extended statistics** command to reset the statistical counters for an IPv6 VRRP-Ev3 session.

```
device# clear ipv6 vrrp-extended statistics
```


Multi-VRF

• Multi-VRF overview.....	501
• Configuring Multi-VRF.....	506

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.

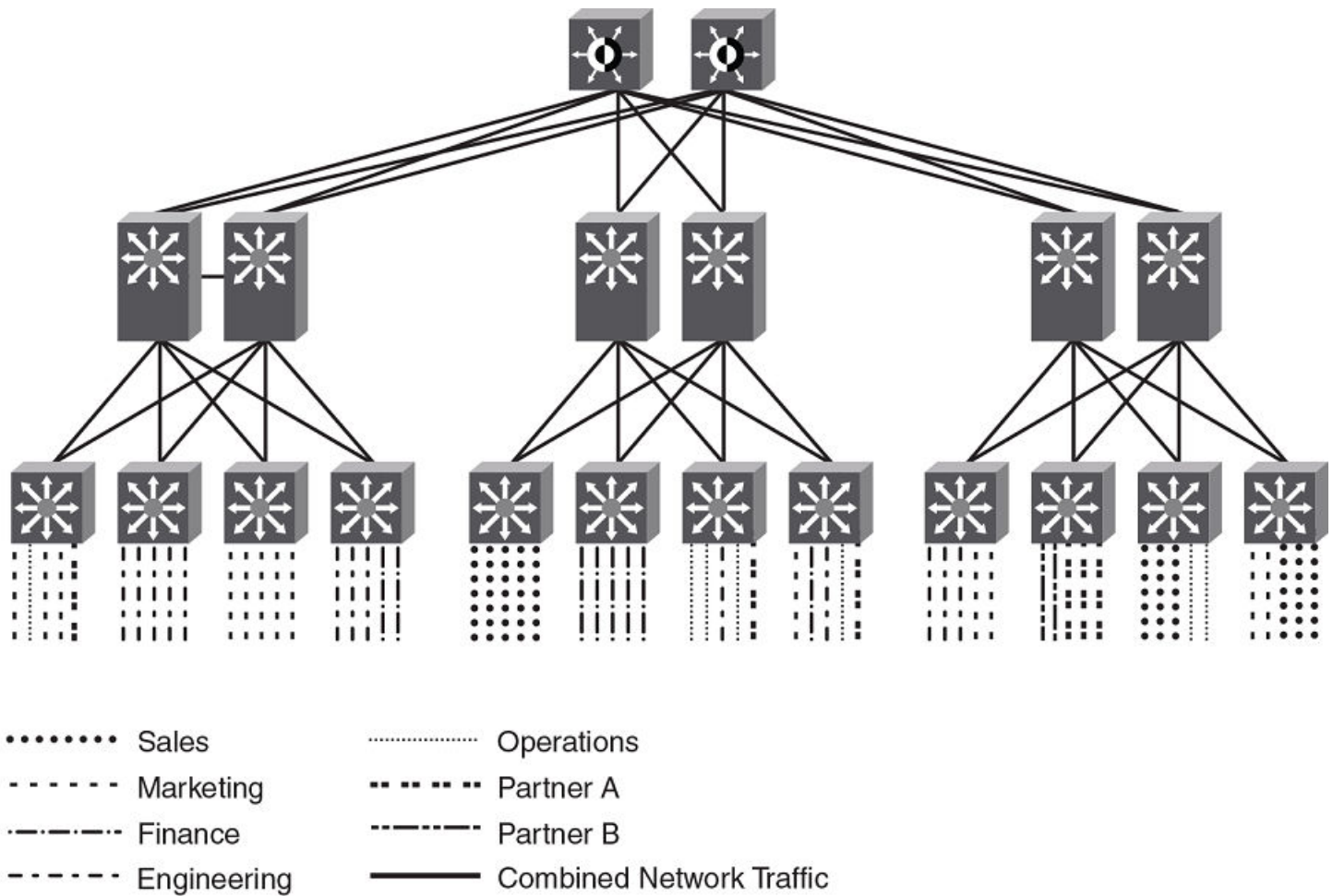
Central to VRF-Lite is the ability to maintain multiple VRF tables on the same Provider Edge (PE) Router. VRF-Lite uses multiple instances of a routing protocol such as OSPF or BGP to exchange route information for a VPN among peer PE routers. The VRF-Lite capable PE router maps an input customer interface 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, RIP, 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 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 41 Example high-level Multi-VRF topology



A Multi-VRF instance can be configured on any of the following:

- Platforms that support untagged physical ports - Applies only to the Brocade ICX 7750 and the Brocade ICX 7450. It is recommended that these ports be configured "route-only" to prevent the leaking of switching traffic if two interfaces in the same VLAN are configured with different VRFs.
- Virtual interfaces
- Loopback interfaces
- Ethernet interfaces
- Tunnel interfaces - The tunnel can belong to any user-defined VRF, but the tunnel source and tunnel destination are restricted to the default VRF.

A Multi-VRF instance **cannot** be configured on any of the following:

- Physical interfaces
- Management interfaces

To configure Multi-VRF, perform the following steps:

- Configure VRF-related system-max values.
- (Optional) Configure tagging on peer interfaces for security.

- Configure VRF instances.
- Configure an IPv4 or IPv6 Address Family (AF) and Neighbor Discovery Protocol for new VRF instances.
- Configure routing protocols for new Multi-VRF instances.
- Assign VRF instances to Layer 3 interfaces.

FastIron considerations for Multi-VRF

When a VRF is configured, a warning message specifies that any configuration existing on the interface is deleted.

When you assign a VRF instance to a static or dynamic LAG, the following rules apply.

- If the LAG is deployed, the primary port can be assigned to a nondefault VRF.
- The dynamic LAG must be configured before any of its ports are assigned to a nondefault VRF routing instance, and all members of the trunk must be in the default VRF.
- Once a dynamic LAG is deployed, all ports are in a LACP blocking state, until the LAG state converges to the forwarding state.
- When a dynamic LAG is undeployed, the primary port remains in the VRF to which it was assigned, but all secondary ports revert to the default VRF.

VRF-related system-max values

The default FastIron configuration does not allow space for VRF routing tables. As a result, you must modify VRF-related system-max values before configuring a VRF instance. The following table lists commands that configure system-max values at the global level.

TABLE 58 Commands for configuring system-max values

Command	Description
ip-vrf	Configures maximum VRF instances supported by the software.
ip-route	Configures maximum IPv4 routes, used to initialize hardware during system init.
ip6-route	Configures maximum IPv6 routes, used to initialize hardware during system init.
ip-route-default-vrf	Configures maximum IPv4 routes to be allocated for the default VRF instance.
ip6-route-default-vrf	Configures maximum IPv6 routes to be allocated for the default VRF instance.
ip-route-vrf	Configures default maximum IPv4 routes to be allocated per user-defined VRF.
ip6-route-vrf	Configures default maximum IPv6 routes to be allocated per user-defined VRF.

This example includes two VRF instances for IPv4 and two VRF instances for IPv6. For the IPv4 partition, the default value for IPv4 TCAM allocation is decreased to 10,000. IPv6 TCAM allocation can then be increased from the default value of 908 to 1408. Both IPv4 and IPv6 VRF instances are planned to allocate 500 routes each.

The following table lists the **ip-vrf** configuration limits for the **system-max** command, by line card and platform.

TABLE 59 Configuration limits for **ip-vrf** with the system-max command

Hardware	Minimum	Default	Maximum
FSX 3rd-generation line card	16	128	128

TABLE 59 Configuration limits for **ip-vrf** with the system-max command (continued)

Hardware	Minimum	Default	Maximum
FSX 2rd-generation line card	0	4	4
ICX 7450	4	32	32
ICX 7750	16	128	128

TABLE 60 Additional configuration limits for the system-max command

Configuration	ICX 7450			ICX 7750		
	Min	Default	Max	Min	Default	Max
ip-route (system-max IPv4 routes that all VRFs in total can support)	4096	12000	15168	98304	98304	131072
ip6-route (system-max IPv6 routes that all VRFs in total can support)	68	5120	5120	5120	5120	7168
ip-route-default-vrf (system-max IPv4 routes configuration for default-VRF)	1024	12000	15168	256	65536	131072
ip-route-vrf (default system-max IPv4 routes per non-default-VRF instances)	128	1024	15168	64	4096	131072
ip6-route-default-vrf (system-max IPv6 routes configuration for default-VRF)	64	5120	5120	64	2048	7168
ip6-route-vrf (default system-max IPv6 routes per non-default-VRF instances, for 3rd generation line cards)	64	100	5120	16	1024	7168

NOTE

The ICX 7250 is not included in the preceding table because the device does not support VRF. The following table provides relevant values for the ICX 7250.

TABLE 61 Configuration limits for ICX 7250 devices

Configuration	Min	Default	Max
ip-route	12000	12000	12000
ip6-route	730	730	730

The following examples illustrate the **system-max** values to support two VRF instances for IPv4 and two instances for IPv6.

- To allocate 2 x 500 routes for IPv4 user-VRF, (10000 - (500+500) = 9000 routes):

```
device(config)# system-max ip-route-default-vrf 9000
Total max configured ipv4 routes are 12000
- Max ipv4 routes configured for default VRF are 9000
- Max ipv4 routes available for all non-default VRFs are 3000
Warning: Please revalidate these values to be valid for your configuration.
Reload required. Please write memory and then reload or power cycle.
device#
```

- To modify the IPv4 partition after modifying the **ip-route-default-vrf** value:

```
device(config)# system-max ip-route 10000
ip-route and ip6-route values changed.
ip-route: 10000
ip6-route: 1408
Warning: Please reconfigure system-max for ip-route-default-vrf and ip-route-vrf (if required).
Reload required. Please write memory and then reload or power cycle.
device#
```


NOTE

This example also modifies the **ip6-route system-max** parameter and is intended only for the ICX 7450.

- To allocate 2 x 500 routes for IPv6 user-VRF ($1408 - (500+500) = 408$):

```
device(config)# system-max ip6-route-default-vrf 408
Total max configured ipv6 routes are 1408
- Max ipv6 routes configured for default VRF are 408
- Max ipv6 routes available for all non-default VRFs are 1000
Warning: Please revalidate these values to be valid for your configuration.
Reload required. Please write memory and then reload or power cycle.
device#
```

- To allocate 500 routes for IPv6 user-VRF:

```
device# system-max ip6-route-vrf 500
Reload required. Please write memory and then reload or power cycle.
device# end
```

- To save the configuration changes:

```
device# write memory
Write startup-config done.
device# Flash Memory Write (8192 bytes per dot) .
Flash to Flash Done.
```

- After the system reloads, the system-max configuration appears as an active configuration.

```
!
system-max ip-route 12000
system-max ip6-route 5120
system-max ip-route-default-vrf 9000
system-max ip6-route-default-vrf 5120
system-max ip-route-vrf 500
system-max ip6-route-vrf 500
!
```

Additional features to support Multi-VRF

In addition to basic features, you can configure dynamic ARP inspection, DHCP snooping, and IP Source Guard to support Multi-VRF.

Static ARP

Static ARP entries help ensure Layer 2 to Layer 3 mappings. This removes some network overhead in the form of ARP requests and replies and can be helpful in managing Multi-VRF networks where devices must communicate on a regular basis. The interface associated with an ARP entry determines which VRF the ARP entry belongs to. However, the additional management involved in adding and maintaining static ARP cache entries must also be taken into account.

An ARP entry is defined by the following parameters:

- IP address
- MAC address
- Type
- Interface

The **arp** command is used to configure static ARP entries on a nondefault VRF interface. (An ARP index is not required before a static ARP is configured.) The **arp** command is available in the address-family mode for a particular VRF.

Proxy ARP

Proxy ARP allows a Layer 3 switch to answer ARP requests from devices on one subnet on behalf of devices in another network. Proxy ARP is configured globally and can be further configured per interface. Interface-level configuration overrides the global configuration.

With the **proxy-arp** command configured, a router does not respond to ARP requests for IP addresses in the same subnet as the incoming ports. The **local-proxy-arp** command permits the router to respond to ARP requests for IP addresses within the same subnet and to forward all traffic between hosts in the subnet. The **local-proxy-arp** command is an interface-level configuration that has no VRF-related impact.

ARP rate limiting

ARP rate limiting is configured globally and applies to all VRFs.

ARP age can be configured globally and on a Layer 3 interface. An ARP age timer configured on a Layer 3 interface overrides the global configuration for ARP aging. The aging timer ensures that the ARP cache does not retain learned entries that are no longer valid.

Dynamic ARP inspection

Dynamic ARP Inspection (DAI) enables the Brocade device to intercept and examine all ARP request and response packets in a subnet and to discard packets with invalid IP-to-MAC address bindings. DAI can prevent common man-in-the-middle (MiM) attacks such as ARP cache poisoning and can prevent the misconfiguration of client IP addresses. DAI allows only valid ARP requests and responses to be forwarded, and supports Multi-VRFs with overlapping address spaces. For more information on DAI, refer to the *FastIron Ethernet Switch Security Configuration Guide*.

DHCP snooping

Dynamic Host Configuration Protocol (DHCP) snooping enables a Brocade device to filter untrusted DHCP IPv4 or IPv6 packets in a subnet. DHCP snooping can ward off MiM attacks, such as a malicious user posing as a DHCP server sending false DHCP server reply packets with the intention of misdirecting other users. DHCP snooping can also stop unauthorized DHCP servers and prevent errors resulting from the user misconfiguration of DHCP servers. DHCP snooping supports Multi-VRFs. For more information on configuring DHCP IPv4 or IPv6 snooping to support a Multi-VRF instance, refer to the *FastIron Ethernet Switch Security Configuration Guide*.

IP Source Guard

You can use IP Source Guard (IPSG) together with DAI on untrusted ports. The Brocade implementation of the IP Source Guard feature supports configuration on a port, on specific VLAN memberships on a port (for Layer 2 devices only), and on specific ports on a virtual Ethernet (VE) interface (for Layer 3 devices only). For more information on IPSG, refer to the *FastIron Ethernet Switch Security Configuration Guide*.

Configuring Multi-VRF

Configuring VRF system-max values

Use this example procedure to modify the default system-max values to accommodate Multi-VRF on a Brocade ICX 7450.

The default **system-max** value must be configured because the device does not have routing table space for user VRFs.

In this example, two user VRFs are configured with 512 maximum routes on each VRF. The *ip-route-default-vrf* and *ip-route-vrf* values must be modified. The **write memory** and **reload** commands are required after the modification.

Once the device has rebooted after the reload, enter the **show default values** command to display the **system-max** settings.

1. Verify the default values.

```
device(config)# show default values
sys log buffers:50          mac age time:300 sec          telnet sessions:5
ip arp age:10 min          bootp relay max hops:4        ip ttl:64 hops
ip addr per intf:24
:
:
System Parameters      Default      Maximum      Current      Configured
ip-arp                 4000         64000        4000         4000
ip-static-arp          512          6000         512          512
pim-mcache             1024         4096         1024         1024
:
:
ip-route               12000        15168        12000        12000
ip-static-route        64           2048         64           64
:
:
ip-vrf                 16           16           16           16
ip-route-default-vrf  12000        15168        12000        12000
ip6-route              5120         5120         5120         5120
ip6-route-default-vr  5120         5120         5120         5120
ip6-route-vrf          100          5120         100          100

device(config)#
```

2. Change the maximum number of routes, save the configuration, and reload the device.

```
device(config)# system-max ip-route-default-vrf 10000
Total max configured ipv4 routes are 12000
- Max ipv4 routes configured for default VRF are 10000
- Max ipv4 routes available for all non-default VRFs are 2000
Warning: Please revalidate these values to be valid for your configuration.
Reload required. Please write memory and then reload or power cycle.
device(config)#
device(config)# system-max ip-route-vrf 512
Reload required. Please write memory and then reload or power cycle.
device(config)#
device(config)# exit
device# write memory
Write startup-config done.
device# Flash Memory Write (8192 bytes per dot) .
Flash to Flash Done.
device# reload
Are you sure? (enter 'y' or 'n'): Rebooting(0)...
y
```

3. Confirm the modified values.

```

device(config)# show default values
sys log buffers:50          mac age time:300 sec          telnet sessions:5
ip arp age:10 min          bootp relay max hops:4        ip ttl:64 hops
ip addr per intf:24
:
:
System Parameters          Default      Maximum      Current      Configured
ip-arp                     4000        64000        4000         4000
ip-static-arp              512         6000         512          512
pim-mcachel               1024        4096         1024         1024
:
:
ip-route                   12000       15168        12000        12000
ip-static-route            64          2048         64           64
:
:
ip-vrf                    16          16           16           16
ip-route-default-vrf      12000       15168        10000        10000
ip6-route                  5120        5120         5120         5120
ip6-route-default-vr      5120        5120         5120         5120
ip6-route-vrf             100         5120         100          100
device(config)#

```

Creating VLANs as links on a tagged port for security

Where Multi-VRF is used, for example, in an enterprise data center, trusted servers or devices should be allowed to communicate directly, and untrusted ones should not be allowed to communicate directly at all. This optional task configures tagged Layer 3 interfaces to support secure VRF instances.

1. In global configuration mode, create a VLAN.

```

device(config)# vlan 10
device(config-vlan-10)#

```

2. Use the **tagged** command to identify the interface as secure.

```

device(config-vlan-10)# tagged e 1/1/1

```

3. Repeat the previous step on the corresponding interface on the peer device.

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. Each VRF instance is identified by a unique Route Distinguisher (RD), which is prepended to the address being advertised. Because the RD provides overlapping client address space with a unique identifier, the same IP address can be used for different VRFs without conflict. The RD can be an AS number, followed by a colon (:) and a unique arbitrary number as shown below. Alternatively, it can be a local IP address followed by a colon (:) and a unique arbitrary number, as in "1.1.1.1:100." 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, IGMP, and multicast for IPv4, 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. (Optional) Assign a Route Distinguisher (RD).

```
device(config-vrf-corporate)# rd 11:11
```

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: 2
Status Codes - A:active, D:pending deletion, I:inactive
Name           Default RD      vrf|v4|v6 Routes Interfaces
corporate      11:11           A | A| I        0
guest          10:10           A | A| I        0
Total number of IPv4 unicast route for all non-default VRF is 0
Total number of IPv6 unicast route for all non-default VRF is 0
```

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 you configure a VRF instance on the device, you must assign one or more Layer 3 interfaces (physical or virtual Ethernet) to the VRF. When you do this, all existing IP addresses are deleted; this action also triggers cache deletion, route deletion, and associated cleanup. After you assign an interface to the VRF, you must reconfigure the IP address and interface properties.

1. Enter global configuration mode.

```
device# configure terminal
```

2. In global configuration mode, enter the **interface ve** command to create a VE interface.

```
device(config)# interface ve 10
```

3. In VE configuration mode, enable forwarding for the VRF "guest".

```
device(config-vif-10)# vrf forwarding guest
Warning: All IPv4 and IPv6 addresses (including link-local) on this interface have been removed
have been removed
```

4. Configure an IPv4 address and mask on the VE interface.

```
device(config-vif-10)# ip address 192.168.1.254/24
```

5. Enable OSPF Area 0.

```
device(config-vif-10)# ip ospf area 0
```

6. Configure the interface as passive.

```
device(config-vif-10)# ip ospf passive
device(config-vif-10)# exit
```

7. Exit the configuration.

```
device(config-vif-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. In global configuration mode, enter interface subtype configuration mode and assign a loopback interface.

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

3. Use the **vrf forwarding** command to assign the interface to the VRF "customer-1" in this example.

```
device(config-lbif-1)# vrf forwarding customer-1
```

4. Assign an IPv4 address and mask to the loopback interface.

```
device(config-lbif-1)# ip address 10.0.0.1/24
```

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: 2
Status Codes - A:active, D:pending deletion, I:inactive
Name Default RD vrf|v4|v6 Routes Interfaces
green 1:1 A | A| A 12 ve111 ve211 ve311*
red 10:12 A | A| A 4 ve1117 port-id tn1*
Total number of IPv4 unicast route for all non-default VRF is 8
Total number of IPv6 unicast route for all non-default VRF is 8
```

To verify a specific VRF in detail mode, enter the **show vrf detail vrf-name** command, as in the following example.

```
device# show vrf green
VRF green, default RD 1:1, Table ID 1
IP Router-Id: 1.1.1.1
Interfaces: ve111 ve211 ve311 ve1116 ve2115
Address Family IPv4
Max Routes: 5500
Number of Unicast Routes: 6
Address Family IPv6
Max Routes: 400
Number of Unicast Routes: 6
```

To verify all configured VRFs in detail mode, enter the **show vrf detail** command, as in the following example.

```
device# show vrf detail
Total number of VRFs configured: 2
VRF green, default RD 1:1, Table ID 1
IP Router-Id: 1.1.1.1
Interfaces: Use "show vrf green" to see the list of interfaces
Address Family IPv4
Max Routes: 5500
Number of Unicast Routes: 6
Address Family IPv6
Max Routes: 400
Number of Unicast Routes: 6
VRF red, default RD 10:12, Table ID 2
IP Router-Id: 1.1.17.1
Interfaces:
Use "show vrf red" to see the list of interfaces
Address Family IPv4
Max Routes: 300
Number of Unicast Routes: 2
Address Family IPv6
Max Routes: 70
Number of Unicast Routes: 2
Total number of IPv4 unicast route for all non-default VRF is 8
Total number of IPv6 unicast route for all non-default VRF is 8
```

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 62 Useful show commands

Default VRF	User-defined VRF
<code>show ip route</code>	<code>show ip route vrf vrf-name</code>
<code>show ip ospf neighbor</code>	<code>show ip ospf vrf vrf-name neighbor</code>
<code>show ip bgp summary</code>	<code>show ip bgp vrf vrf-name summary</code>

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** 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(config-if-e1000-1/7/1)# no vrf forwarding1
All existing IP and IPv6 address will be removed from port 1/7/1
The port will be returned to default VRF
```

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
device(config-vrf-customer1)#
```

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
Warning: All IPv4 and IPv6 addresses (including link-local) from all interfaces in VRF customer1 have been removed
```

Configuring static ARP for Multi-VRF

The interface associated with an ARP entry determines to which VRF the ARP entry belongs.

An ARP entry is defined by the following parameters:

- IP address
- MAC address
- Type
- Interface

1. The following example illustrates how to configure static ARP on default VRFs on an Ethernet interface.

```
device(config)# arp 192.168.1.100 0000.2344.2441 eth 1/7/1
```


2. The following example illustrates how to configure static ARP on nondefault VRFs.

NOTE

The **arp** command can be used to configure static-ARP entries on a nondefault VRF interface. The VRF command does not require an ARP index before a static-ARP is configured. The **arp** command is available in the address-family mode for a particular VRF.

```
device(config)#
device(config)# vrf customer-1
device(config-vrf-customer-1)# address-family ipv4
device(config-vrf-customer-1-ipv4)# arp 1.1.1.1 0004.8044.5566 ethernet 1/7/8
device(config-vrf-customer-1-ipv4)# exit-address-family
device(config-vrf-customer-1)# exit-vrf
device(config)#
```

Configuring additional ARP features for Multi-VRF

This section discusses options for configuring proxy ARP and ARP rate limiting.

Proxy ARP allows a Layer 3 switch to answer ARP requests from devices on one subnet on behalf of devices in another network. Proxy ARP is configured globally and can be further configured per interface. Interface-level configuration overrides the global configuration.

With the **proxy-arp** command configured, a router does not respond to ARP requests for IP addresses in the same subnet as the incoming ports. The **local-proxy-arp** command permits the router to respond to ARP requests for IP addresses within the same subnet and to forward all traffic between hosts in the subnet. The **local-proxy-arp** command is an interface-level configuration that has no VRF-related impact.

ARP rate limiting is configured globally and applies to all VRFs.

ARP age can be configured globally and on a Layer 3 interface. An ARP age timer configured on a Layer 3 interface overrides the global configuration for ARP aging. The aging timer ensures that the ARP cache does not retain learned entries that are no longer valid.

To configure proxy ARP globally:

```
device(config)# proxy-arp
```

To configure proxy ARP on a Layer 3 Ethernet interface:

```
device(config)# int e1000 1/7/1
device(config-if-e1000-1/7/1)# local-proxy-arp
```

To configure ARP rate limiting globally:

```
device(config)# rate-limit-arp
```

To configure ARP rate limiting on a Layer 3 Ethernet interface for an aging timeout of 20 minutes:

```
device(config)# int e1000 1/7/1
device(config-if-e1000-1/7/1)# ip arp-age 20
```