



Routing Configuration Guide for Cisco 8000 Series Routers, IOS XR Release 7.5.x

First Published: 2021-11-30

Americas Headquarters

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



CONTENTS

PREFACE

Preface xv

Changes to This Document xv

Communications, Services, and Additional Information xv

CHAPTER 1

Implementing IS-IS 1

Prerequisites for Implementing IS-IS 2

Restrictions for Implementing IS-IS 2

Information About Implementing IS-IS 2

IS-IS Functional Overview 2

Key Features Supported in the Cisco IOS XR IS-IS Implementation 3

IS-IS Configuration Grouping 3

Router Configuration Mode 3

Router Address Family Configuration Mode 4

Interface Configuration Mode 4

Interface Address Family Configuration Mode 4

Multitopology Configuration 4

Limit LSP Flooding 4

Control LSP Flooding for IS-IS 5

IPv6 Routing and Configuring IPv6 Addressing 9

Flood Blocking on Specific Interfaces 9

Maximum LSP Lifetime and Refresh Interval 9

Minimum Remaining Lifetime 9

Mesh Group Configuration 10

Multitopology IPv6 for IS-IS 10

IS-IS Authentication 11

Configure Authentication for IS-IS 11

Configure Keychains for IS-IS	13
Multi-Instance IS-IS	14
Enable IS-IS and Configure Level 1 or Level 2 Routing	15
Single-Topology IPv6	16
Configure Single Topology for IS-IS	17
Set SPF Interval for a Single-Topology Configuration	21
Customize Routes for IS-IS	23
Set Priority for Adding Prefixes to RIB	27
IS-IS Interfaces	28
Tag IS-IS Interface Routes	28
Nonstop Forwarding	31
Configure Nonstop Forwarding for IS-IS	32
ISIS NSR	34
Configuring ISIS-NSR	34
Configuring IS-IS Adjacency Stagger	36
Multiprotocol Label Switching Traffic Engineering	36
Configure MPLS Traffic Engineering for IS-IS	37
MPLS TE Forwarding Adjacency	39
Tune Adjacencies for IS-IS	39
MPLS Label Distribution Protocol IGP Synchronization	42
Configuring MPLS LDP IS-IS Synchronization	42
Overload Bit on Router	44
Overload Bit Configuration During Multitopology Operation	44
IS-IS Overload Bit Avoidance	44
Configure IS-IS Overload Bit Avoidance	45
Default Routes	46
Attached Bit on an IS-IS Instance	46
IS-IS Support for Route Tags	46
Multicast-Intact Feature	46
Multicast Topology Support Using IS-IS	47
MPLS TE Interarea Tunnels	47
IP Fast Reroute	47
Unequal Cost Multipath Load-balancing for IS-IS	48
Configuring Multitopology Routing	48

Restrictions for Configuring Multitopology Routing	49
Information About Multitopology Routing	49
Configuring a Global Topology and Associating It with an Interface	49
Enabling an IS-IS Topology	51
Placing an Interface in a Topology in IS-IS	52
Configuring a Routing Policy	53
Configuring Multitopology for IS-IS	54
Enabling Multicast-Intact	54
Configuring IP/LDP Fast Reroute	55
ISIS Link Group	58
Configure Link Group Profile	58
Configure Link Group Interface	61
Configuration Examples for Implementing IS-IS	62
Configuring Single-Topology IS-IS for IPv6: Example	63
Configuring Multitopology IS-IS for IPv6: Example	63
Redistributing IS-IS Routes Between Multiple Instances: Example	63
Tagging Routes: Example	64
Configuring IS-IS Overload Bit Avoidance: Example	64
Example: Configuring IS-IS To Handle Router Overload	65
Configuring Global Weighted SRLG Protection	70
Label Distribution Protocol IGP Auto-configuration	72
MPLS LDP-IGP Synchronization Compatibility with LDP Graceful Restart	72
MPLS LDP-IGP Synchronization Compatibility with IGP Nonstop Forwarding	72
Management Information Base (MIB) for IS-IS	73
Support for a Configurable Knob to Reject ISIS PDU on L2 Interfaces	73

CHAPTER 2
Implementing OSPF 79

Prerequisites for Implementing OSPF	79
Information About Implementing OSPF	80
OSPF Functional Overview	80
Key Features Supported in the Cisco IOS XR Software OSPF Implementation	81
Comparison of Cisco IOS XR Software OSPFv3 and OSPFv2	82
OSPF Routing Components	82
Autonomous Systems	83

Areas	83
Routers	84
Area Border Routers	84
Autonomous System Boundary Routers (ASBR)	84
Interior Routers	85
OSPF Process and Router ID	85
Supported OSPF Network Types	85
Route Authentication Methods for OSPF	86
Plain Text Authentication	86
MD5 Authentication	86
Authentication Strategies	86
Key Rollover	86
Neighbors and Adjacency for OSPF	87
OSPF strict-mode Support for BFD Dampening	87
Enabling strict-mode	87
BFD strict-mode: Example	88
OSPF FIB Download Notification	89
Designated Router (DR) for OSPF	89
Default Route for OSPF	90
Link-State Advertisement Types for OSPF Version 2	90
Link-State Advertisement Types for OSPFv3	91
Virtual Link and Transit Area for OSPF	92
Passive Interface	93
OSPFv2 Sham Link Support for MPLS VPN	93
OSPFv3 Sham Link Support for MPLS VPN	94
Graceful Restart Procedure over the Sham-link	95
ECMP and OSPFv3 Sham-link	95
OSPFv2OSPF SPF Prefix Prioritization	95
Route Redistribution for OSPF	96
OSPF Shortest Path First Throttling	96
Nonstop Forwarding for OSPF Version 2	97
Graceful Shutdown for OSPFv3	98
Modes of Graceful Restart Operation	98
Warm Standby and Nonstop Routing for OSPF Version 2	101

Warm Standby and Nonstop Routing for OSPF Version 3	102
Multicast-Intact Support for OSPF	102
Load Balancing in OSPF Version 2 and OSPFv3	102
Configure Prefix Suppression for OSPF	103
Configure Prefix Suppression for OSPFv3	107
Multi-Area Adjacency for OSPF Version 2	112
Label Distribution Protocol IGP Auto-configuration for OSPF	113
OSPF Authentication Message Digest Management	113
GTSM TTL Security Mechanism for OSPF	113
Path Computation Element for OSPFv2	114
OSPF IP Fast Reroute Loop Free Alternate	114
Management Information Base (MIB) for OSPFv3	114
Management Information Base (MIB) for OSPFv2	115
VRF-lite Support for OSPFv2	115
OSPFv3 Timers Update	115
Unequal Cost Multipath Load-balancing for OSPF	115
How to Implement OSPF	116
Enable OSPF	116
Configuring Stub and Not-So-Stubby Area Types	119
Configuring Neighbors for Nonbroadcast Networks	121
Configuring Authentication at Different Hierarchical Levels for OSPF Version 2	124
Control Frequency That Same LSA Is Originated or Accepted for OSPF	127
Creating a Virtual Link with MD5 Authentication to Area 0 for OSPF	129
Redistribute Routes into OSPF	132
Configuring OSPF Shortest Path First Throttling	135
Examples	136
Configuring Nonstop Forwarding Specific to Cisco for OSPF Version 2	137
Configuring OSPF Version 2 for MPLS Traffic Engineering	139
Configuring OSPFv3 Graceful Restart	141
Configuring an OSPFv2 Sham Link	143
Configure OSPFv2 OSPF SPF Prefix Prioritization	146
Enabling Multicast-intact for OSPFv2	148
Associating Interfaces to a VRF	149
Configuring OSPF as a Provider Edge to Customer Edge (PE-CE) Protocol	150

Creating Multiple OSPF Instances (OSPF Process and a VRF)	152
Configuring Multi-area Adjacency	154
Configuring Label Distribution Protocol IGP Auto-configuration for OSPF	156
Configuring LDP IGP Synchronization: OSPF	156
Configuring Authentication Message Digest Management for OSPF	158
Examples	159
Configuring Generalized TTL Security Mechanism (GTSM) for OSPF	160
Examples	162
Verify OSPF Configuration and Operation	163
Configuring IP Fast Reroute Loop-free Alternate	165
Enabling IPFRR LFA	165
Excluding an Interface From IP Fast Reroute Per-link Computation	166
Enabling OSPF Interaction with SRMS Server	167
Configuration Examples for Implementing OSPF	169
Cisco IOS XR Software for OSPF Version 2 Configuration: Example	169
CLI Inheritance and Precedence for OSPF Version 2: Example	170
MPLS TE for OSPF Version 2: Example	171
ABR with Summarization for OSPFv3: Example	171
ABR Stub Area for OSPFv3: Example	172
ABR Totally Stub Area for OSPFv3: Example	172
Configuring OSPFv2 SPF Prefix Prioritization: Example	172
Route Redistribution for OSPFv3: Example	173
Virtual Link Configured Through Area 1 for OSPFv3: Example	173
Virtual Link Configured with MD5 Authentication for OSPF Version 2: Example	174
VPN Backbone and Sham Link Configured for OSPF Version 2: Example	175
Enable OSPF	177
Verify OSPF Configuration and Operation	180
Control Frequency That Same LSA Is Originated or Accepted for OSPF	182
Create Virtual Link	184
Summarize Subnetwork LSAs on OSPF ABR	189
Graceful Shutdown for OSPFv3	191
Enable Nonstop Routing for OSPFv2	192
Warm Standby and Nonstop Routing for OSPF Version 3	193
Enable Nonstop Routing for OSPFv3	193

OSPFv2/OSPF SPF Prefix Prioritization	194
Configuring OSPFv2 SPF Prefix Prioritization: Example	195
OSPFv3 Timers Update	196
OSPF Link-State Database Overload Protection	196

CHAPTER 3**Implementing and Monitoring RIB 199**

Prerequisites for Implementing RIB	199
Information About RIB Configuration	200
Overview of RIB	200
RIB Data Structures in BGP and Other Protocols	200
RIB Administrative Distance	200
RIB Support for IPv4	201
RIB Statistics	201
RIB Quarantining	202
Route Consistency Checker	202
How to Deploy and Monitor RIB	203
Verifying RIB Configuration Using the Routing Table	203
Verifying Networking and Routing Problems	203
Disabling RIB Next-hop Dampening	205
BGP-RIB Feedback Mechanism for Update Generation	205
Configuration Examples for RIB Monitoring	206
Output of show route Command: Example	206
Output of show route backup Command: Example	206
Output of show route best-local Command: Example	207
Output of show route connected Command: Example	207
Output of show route local Command: Example	207
Output of show route longer-prefixes Command: Example	207
Output of show route next-hop Command: Example	208
Enabling RCC: Example	208

CHAPTER 4**Implementing Routing Policy 209**

Restrictions for Implementing Routing Policy	209
Define Route Policy	210
Attach Routing Policy to BGP Neighbor	211

Modify Routing Policy Using Text Editor	212
References for Routing Policy	215
Routing Policy Language	215
Routing Policy Language Overview	216
Routing Policy Language Structure	216
Routing Policy Language Components	223
Routing Policy Language Usage	224
Policy Definitions	226
Parameterization	227
Parameterization at Attach Points	227
Global Parameterization	228
Semantics of Policy Application	228
Boolean Operator Precedence	229
Multiple Modifications of Same Attribute	229
When Attributes Are Modified	230
Default Drop Disposition	230
Control Flow	231
Policy Verification	231
Range Checking	232
Incomplete Policy and Set References	232
Aggregation	232
Policy Statements	233
Remark	233
Disposition	234
Action	236
If	236
Boolean Conditions	237
apply	238
Attach Points	239
BGP Policy Attach Points	239
OSPF Policy Attach Points	259
OSPFv3 Policy Attach Points	262
IS-IS Policy Attach Points	264
Nondestructive Editing of Routing Policy	265

Attached Policy Modification	265
Nonattached Policy Modification	265
Editing Routing Policy Configuration Elements	266
Hierarchical Policy Conditions	268
Apply Condition Policies	268
Nested Wildcard Apply Policy	270
VRF Import Policy Enhancement	271
Match Aggregated Route	271
Remove Private AS in Inbound Policy	271

CHAPTER 5**Implementing Static Routes 273**

Prerequisites for Implementing Static Routes	273
Restrictions for Implementing Static Routes	273
Information About Implementing Static Routes	274
Static Route Functional Overview	274
Default Administrative Distance	274
Directly Connected Routes	275
Recursive Static Routes	275
Fully Specified Static Routes	276
Floating Static Routes	276
Default VRF	276
IPv4 and IPv6 Static VRF Routes	277
Dynamic ECMP	277
How to Implement Static Routes	277
Configure Static Route	277
Configure Floating Static Route	279
Configure Static Routes Between PE-CE Routers	280
Change Maximum Number of Allowable Static Routes	282
Associate VRF with a Static Route	283
Configuration Examples	284
Configure Native UCMP for Static Routing	284
IPv4 Multicast Static Routes	286
Configure Multicast Static Routes	286
References for Static Routes	287

Static Route Functional Overview	288
Default Administrative Distance	289
Directly Connected Routes	289
Floating Static Routes	289
Fully Specified Static Routes	289
Recursive Static Routes	290

CHAPTER 6

Implementing UCMP 291

ECMP vs. UCMP Load Balancing	292
UCMP Minimum Integer Ratio	292
Configuring IS-IS With Weight	293
Configuring IS-IS With Metric	294
Configuring BGP With Weights	295

CHAPTER 7

Implementing BFD 297

Prerequisites for Implementing BFD	297
Restrictions for Implementing BFD	298
Information About BFD	298
BFD Packet Intervals on Physical Interfaces	298
Control Packet Failure Detection In Asynchronous Mode	298
Priority Settings for BFD Packets	298
BFD over Bundles IETF Mode Support on a Per Bundle Basis	299
BFD Dampening	299
BFD Hardware Offload Support for IPv4	300
BFD Hardware Offload Support for IPv6	301
IPv4 Multihop BFD	302
Configure BFD	303
Configure BFD Under a Dynamic Routing Protocol or Use a Static Route	303
Enabling BFD on a BGP Neighbor	303
Enabling BFD for OSPF on an Interface	305
Enabling BFD on a Static Route	305
Specifying the BFD Destination Address on a Bundle	306
Enabling BFD Sessions on Bundle Members	307
Configuring the Minimum Thresholds for Maintaining an Active Bundle	308

Configuring BFD Packet Transmission Intervals and Failure Detection Times on a Bundle	308
Configure BFD over Bundles IETF Mode Support on a Per Bundle Basis	309
Enabling Echo Mode to Test the Forwarding Path to a BFD Peer	310
Overriding the Default Echo Packet Source Address	310
Specifying the Echo Packet Source Address Globally for BFD	310
Specifying the Echo Packet Source Address on an Individual Interface	311
Disabling Echo Mode	312
Disabling Echo Mode on a Router	312
Disabling Echo Mode on an Individual Interface	312
Minimizing BFD Session Flapping Using BFD Dampening	313
Clear and Display BFD Counters	314
BFD IPv6 in Bundle Manager Domain	314
BFD Over BGP: Example	314
BFD Over OSPF: Example	314
BFD Over Static Routes: Example	315
BFD Echo Mode Disable: Examples	315
Echo Packet Source Address: Examples	315
BFD Dampening: Examples	316
BFD Hardware Offload for RSVP Tail-End	316
BFD over MPLS Traffic Engineering LSPs	316
Configuring BFD over MPLS Traffic Engineering LSPs	317
Enabling BFD Parameters for BFD over TE Tunnels	317
Configuring BFD Bring up Timeout	318
Configuring BFD Dampening for TE Tunnels	318
Configuring Periodic LSP Ping Requests	319
Configuring BFD at the Tail-End	320
Configuring BFD over LSP Sessions on Line Cards	320
BFD over MPLS TE Tunnel Tail-End Configuration	321
Configuration Examples for Configuring BFD	322
BFD over MPLS TE LSPs Examples	322
BFD Over MPLS TE Tunnel Head-End Configuration: Example	322
BFD Over MPLS TE Tunnel Tail-End Configuration: Example	322
RFCs	322
Technical Assistance	322

CHAPTER 8

Implementing Fast Reroute Loop-Free Alternate	323
Prerequisites for IPv4/IPv6 Loop-Free Alternate Fast Reroute	323
Restrictions for Loop-Free Alternate Fast Reroute	323
IS-IS and IP FRR	324
Repair Paths	324
LFA Overview	324
LFA Calculation	325
Interaction Between RIB and Routing Protocols	325
Fast Reroute with Remote Loop-Free Alternate	326
Configuration	327
Running Configuration	327
Verification	329
Configuring IPv4 or IPv6 Loop-Free Alternate Fast Reroute Support: Example	330



Preface

This preface contains these sections:

- [Changes to This Document, on page xv](#)
- [Communications, Services, and Additional Information, on page xv](#)

Changes to This Document

Table 1: Changes to This Document

Date	Change Summary
November 2021	Initial release of this document.

Communications, Services, and Additional Information

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

Cisco Bug Search Tool

[Cisco Bug Search Tool](#) (BST) is a web-based tool that acts as a gateway to the Cisco bug tracking system that maintains a comprehensive list of defects and vulnerabilities in Cisco products and software. BST provides you with detailed defect information about your products and software.



CHAPTER 1

Implementing IS-IS

Integrated Intermediate System-to-Intermediate System (IS-IS), Internet Protocol Version 4 (IPv4), is a standards-based Interior Gateway Protocol (IGP). The Cisco software implements the IP routing capabilities described in International Organization for Standardization (ISO)/International Engineering Consortium (IEC) 10589 and RFC 1195, and adds the standard extensions for single topology and multitopology IS-IS for IP Version 6 (IPv6).

This module describes how to implement IS-IS (IPv4 and IPv6) on your Cisco IOS XR network.

- [Prerequisites for Implementing IS-IS, on page 2](#)
- [Restrictions for Implementing IS-IS, on page 2](#)
- [Information About Implementing IS-IS , on page 2](#)
- [Multitopology Configuration, on page 4](#)
- [Limit LSP Flooding, on page 4](#)
- [IPv6 Routing and Configuring IPv6 Addressing, on page 9](#)
- [Flood Blocking on Specific Interfaces, on page 9](#)
- [Multitopology IPv6 for IS-IS, on page 10](#)
- [IS-IS Authentication, on page 11](#)
- [Multi-Instance IS-IS, on page 14](#)
- [Enable IS-IS and Configure Level 1 or Level 2 Routing, on page 15](#)
- [Single-Topology IPv6, on page 16](#)
- [Customize Routes for IS-IS, on page 23](#)
- [Set Priority for Adding Prefixes to RIB, on page 27](#)
- [IS-IS Interfaces, on page 28](#)
- [Nonstop Forwarding, on page 31](#)
- [ISIS NSR, on page 34](#)
- [Multiprotocol Label Switching Traffic Engineering, on page 36](#)
- [Overload Bit on Router, on page 44](#)
- [IS-IS Overload Bit Avoidance, on page 44](#)
- [Default Routes, on page 46](#)
- [Attached Bit on an IS-IS Instance, on page 46](#)
- [IS-IS Support for Route Tags, on page 46](#)
- [Multicast-Intact Feature , on page 46](#)
- [Multicast Topology Support Using IS-IS, on page 47](#)
- [MPLS TE Interarea Tunnels , on page 47](#)
- [IP Fast Reroute, on page 47](#)

- [Unequal Cost Multipath Load-balancing for IS-IS, on page 48](#)
- [Configuring Multitopology Routing, on page 48](#)
- [Restrictions for Configuring Multitopology Routing, on page 49](#)
- [Information About Multitopology Routing, on page 49](#)
- [Configuring a Global Topology and Associating It with an Interface, on page 49](#)
- [Enabling an IS-IS Topology, on page 51](#)
- [Placing an Interface in a Topology in IS-IS, on page 52](#)
- [Configuring a Routing Policy, on page 53](#)
- [Configuring Multitopology for IS-IS, on page 54](#)
- [Enabling Multicast-Intact , on page 54](#)
- [Configuring IP/LDP Fast Reroute , on page 55](#)
- [ISIS Link Group , on page 58](#)
- [Configure Link Group Profile, on page 58](#)
- [Configure Link Group Interface, on page 61](#)
- [Configuration Examples for Implementing IS-IS , on page 62](#)
- [Configuring Global Weighted SRLG Protection, on page 70](#)
- [Label Distribution Protocol IGP Auto-configuration, on page 72](#)
- [Management Information Base \(MIB\) for IS-IS, on page 73](#)
- [Support for a Configurable Knob to Reject ISIS PDU on L2 Interfaces, on page 73](#)

Prerequisites for Implementing IS-IS

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Restrictions for Implementing IS-IS

When multiple instances of IS-IS are being run, an interface can be associated with only one instance (process). Instances may not share an interface.

Information About Implementing IS-IS

To implement IS-IS you need to understand the following concepts:

IS-IS Functional Overview

Small IS-IS networks are typically built as a single area that includes all routers in the network. As the network grows larger, it may be reorganized into a backbone area made up of the connected set of all Level 2 routers from all areas, which is in turn connected to local areas. Within a local area, routers know how to reach all system IDs. Between areas, routers know how to reach the backbone, and the backbone routers know how to reach other areas.

The IS-IS routing protocol supports the configuration of backbone Level 2 and Level 1 areas and the necessary support for moving routing information between the areas. Routers establish Level 1 adjacencies to perform

routing within a local area (intra-area routing). Routers establish Level 2 adjacencies to perform routing between Level 1 areas (interarea routing).

Each IS-IS instance can support either a single Level 1 or Level 2 area, or one of each. By default, all IS-IS instances automatically support Level 1 and Level 2 routing. You can change the level of routing to be performed by a particular routing instance using the **is-type** command.

Restrictions

When multiple instances of IS-IS are being run, an interface can be associated with only one instance (process). Instances may not share an interface.

Key Features Supported in the Cisco IOS XR IS-IS Implementation

The Cisco IOS XR implementation of IS-IS conforms to the IS-IS Version 2 specifications detailed in RFC 1195 and the IPv6 IS-IS functionality based on the Internet Engineering Task Force (IETF) IS-IS Working Group draft-ietf-isis-ipv6.txt document.

The following list outlines key features supported in the Cisco IOS XR implementation:

- Single topology IPv6
- Multitopology
- Nonstop forwarding (NSF), both Cisco proprietary and IETF
- Three-way handshake
- Mesh groups
- Multiple IS-IS instances
- Configuration of a broadcast medium connecting two networking devices as a point-to-point link
- Fast-flooding with different threads handling flooding and shortest path first (SPF).



Note For information on IS-IS support for Bidirectional Forwarding Detection (BFD), see and .

IS-IS Configuration Grouping

Cisco IOS XR groups all of the IS-IS configuration in router IS-IS configuration mode, including the portion of the interface configurations associated with IS-IS. To display the IS-IS configuration in its entirety, use the **show running router isis** command. The command output displays the running configuration for all configured IS-IS instances, including the interface assignments and interface attributes.

Router Configuration Mode

The following example shows how to enter router configuration mode:

```
RP/0/# configuration
RP/0/(config)# router isis isp
```

```
RP/0/(config-isis)#
```

Router Address Family Configuration Mode

The following example shows how to enter router address family configuration mode:

```
RP/0/(config)# router isis isp
RP/0/(config-isis)# address-family
ipv4 u
nicast
RP/0/(config-isis-af)#
```

Interface Configuration Mode

The following example shows how to enter interface configuration mode:

```
RP/0/(config)# router isis isp
RP/0/(config-isis)# interface GigabitEthernet 0
/3/0/0
RP/0/(config-isis-if)#
```

Interface Address Family Configuration Mode

The following example shows how to enter interface address family configuration mode:

```
RP/0/(config)# router isis isp
RP/0/(config-isis)# interface
GigabitEthernet 0 /3/0/0
RP/0/(config-isis-if)# address-family ipv4 unicast
RP/0/(config-isis-if-af)#
```

Multitopology Configuration

The software supports multitopology for IPv6 IS-IS unless single topology is explicitly configured in IPv6 address-family configuration mode.



Note IS-IS supports IP routing and not Open Systems Interconnection (OSI) Connectionless Network Service (CLNS) routing.

Limit LSP Flooding

Limiting link-state packets (LSP) may be desirable in certain “meshy” network topologies. An example of such a network might be a highly redundant one such as a fully meshed set of point-to-point links over a

nonbroadcast multiaccess (NBMA) transport. In such networks, full LSP flooding can limit network scalability. One way to restrict the size of the flooding domain is to introduce hierarchy by using multiple Level 1 areas and a Level 2 area. However, two other techniques can be used instead of or with hierarchy: Block flooding on specific interfaces and configure mesh groups.

Both techniques operate by restricting the flooding of LSPs in some fashion. A direct consequence is that although scalability of the network is improved, the reliability of the network (in the face of failures) is reduced because a series of failures may prevent LSPs from being flooded throughout the network, even though links exist that would allow flooding if blocking or mesh groups had not restricted their use. In such a case, the link-state databases of different routers in the network may no longer be synchronized. Consequences such as persistent forwarding loops can ensue. For this reason, we recommend that blocking or mesh groups be used only if specifically required, and then only after careful network design.

Control LSP Flooding for IS-IS

Flooding of LSPs can limit network scalability. You can control LSP flooding by tuning your LSP database parameters on the router globally or on the interface. This task is optional.

Many of the commands to control LSP flooding contain an option to specify the level to which they apply. Without the option, the command applies to both levels. If an option is configured for one level, the other level continues to use the default value. To configure options for both levels, use the command twice. For example:

```
RP/0/RP0/CPU0:router(config-isis)# lsp-refresh-interval 1200 level 2
RP/0/RP0/CPU0:router(config-isis)# lsp-refresh-interval 1100 level 1
```

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **lsp-refresh-interval** *seconds* [**level** { **1** | **2** }]
4. **lsp-check-interval** *seconds* [**level** { **1** | **2** }]
5. **lsp-gen-interval** { [**initial-wait** *initial* | **secondary-wait** *secondary* | **maximum-wait** *maximum*] ... } [**level** { **1** | **2** }]
6. **lsp-mtu** *bytes* [**level** { **1** | **2** }]
7. **max-lsp-lifetime** *seconds* [**level** { **1** | **2** }]
8. **ignore-lsp-errors** **disable**
9. **interface** *type interface-path-id*
10. **lsp-interval** *milliseconds* [**level** { **1** | **2** }]
11. **csnp-interval** *seconds* [**level** { **1** | **2** }]
12. **retransmit-interval** *seconds* [**level** { **1** | **2** }]
13. **retransmit-throttle-interval** *milliseconds* [**level** { **1** | **2** }]
14. **mesh-group** { *number* | **blocked** }
15. Use the **commit** or **end** command.
16. **show isis** **interface** [*type interface-path-id* | **level** { **1** | **2** }] [**brief**]
17. **show isis** [**instance** *instance-id*] **database** [**level** { **1** | **2** }] [**detail** | **summary** | **verbose**] [* | *lsp-id*]
18. **show isis** [**instance** *instance-id*] **lsp-log** [**level** { **1** | **2** }]

19. show isis database-log [level { 1 | 2 }]**DETAILED STEPS****Step 1** **configure****Example:**

```
RP/0/# configure
```

Enters mode.

Step 2 **router isis** *instance-id***Example:**

```
RP/0/RP0/CPU0:router(config)# router isis isp
```

Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.

- You can change the level of routing to be performed by a particular routing instance by using the **is-type** router configuration command.

Step 3 **lsp-refresh-interval** *seconds* [**level** { 1 | 2 }]**Example:**

```
RP/0/RP0/CPU0:router(config-isis)# lsp-refresh-interval 10800
```

(Optional) Sets the time between regeneration of LSPs that contain different sequence numbers

- The refresh interval should always be set lower than the **max-lsp-lifetime** command.

Step 4 **lsp-check-interval** *seconds* [**level** { 1 | 2 }]**Example:**

```
RP/0/RP0/CPU0:router(config-isis)# lsp-check-interval 240
```

(Optional) Configures the time between periodic checks of the entire database to validate the checksums of the LSPs in the database.

- This operation is costly in terms of CPU and so should be configured to occur infrequently.

Step 5 **lsp-gen-interval** { [**initial-wait** *initial* | **secondary-wait** *secondary* | **maximum-wait** *maximum*] ... } [**level** { 1 | 2 }]**Example:**

```
RP/0/RP0/CPU0:router(config-isis)# lsp-gen-interval maximum-wait 15 initial-wait 5
```

(Optional) Reduces the rate of LSP generation during periods of instability in the network. Helps reduce the CPU load on the router and number of LSP transmissions to its IS-IS neighbors.

- During prolonged periods of network instability, repeated recalculation of LSPs can cause an increased CPU load on the local router. Further, the flooding of these recalculated LSPs to the other Intermediate Systems in the network causes increased traffic and can result in other routers having to spend more time running route calculations.

Step 6 **lsp-mtu** *bytes* [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis)# lsp-mtu 1300
```

(Optional) Sets the maximum transmission unit (MTU) size of LSPs.

Step 7 **max-lsp-lifetime** *seconds* [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis)# max-lsp-lifetime 11000
```

(Optional) Sets the initial lifetime given to an LSP originated by the router.

- This is the amount of time that the LSP persists in the database of a neighbor unless the LSP is regenerated or refreshed.

Step 8 **ignore-lsp-errors** **disable**

Example:

```
RP/0/RP0/CPU0:router(config-isis)# ignore-lsp-errors disable
```

(Optional) Sets the router to purge LSPs received with checksum errors.

Step 9 **interface** *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router(config-isis)# interface HundredGigE 0/1/0/3
```

Enters interface configuration mode.

Step 10 **lsp-interval** *milliseconds* [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis-if)# lsp-interval 100
```

(Optional) Configures the amount of time between each LSP sent on an interface.

Step 11 **csnp-interval** *seconds* [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis-if)# csnp-interval 30 level 1
```

(Optional) Configures the interval at which periodic CSNP packets are sent on broadcast interfaces.

- Sending more frequent CSNPs means that adjacent routers must work harder to receive them.
- Sending less frequent CSNP means that differences in the adjacent routers may persist longer.

Step 12 **retransmit-interval** *seconds* [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis-if)# retransmit-interval 60
```

(Optional) Configures the amount of time that the sending router waits for an acknowledgment before it considers that the LSP was not received and subsequently resends.

```
RP/0/RP0/CPU0:router(config-isis-if)# retransmit-interval 60
```

Step 13 **retransmit-throttle-interval** *milliseconds* [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis-if)# retransmit-throttle-interval 1000
```

(Optional) Configures the amount of time between retransmissions on each LSP on a point-to-point interface.

- This time is usually greater than or equal to the **lsp-interval** command time because the reason for lost LSPs may be that a neighboring router is busy. A longer interval gives the neighbor more time to receive transmissions.

Step 14 **mesh-group** { *number* | **blocked** }

Example:

```
RP/0/RP0/CPU0:router(config-isis-if)# mesh-group blocked
```

(Optional) Optimizes LSP flooding in NBMA networks with highly meshed, point-to-point topologies.

- This command is appropriate only for an NBMA network with highly meshed, point-to-point topologies.

Step 15 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 16 **show isis interface** [*type interface-path-id* | **level** { **1** | **2** }] [**brief**]

Example:

```
RP/0/RP0/CPU0:router# show isis interface HundredGigE 0/1/0/1 brief
```

(Optional) Displays information about the IS-IS interface.

Step 17 **show isis** [**instance** *instance-id*] **database** [**level** { **1** | **2** }] [**detail** | **summary** | **verbose**] [* | *lsp-id*]

Example:

```
RP/0/RP0/CPU0:router# show isis database level 1
```

(Optional) Displays the IS-IS LSP database.

Step 18 **show isis** [**instance** *instance-id*] **lsp-log** [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router# show isis lsp-log
```


(Optional) Displays LSP log information.

Step 19 `show isis database-log [level { 1 | 2 }]`

Example:

```
RP/0/RP0/CPU0:router# show isis database-log level 1
```

(Optional) Display IS-IS database log information.

IPv6 Routing and Configuring IPv6 Addressing

By default, IPv6 routing is disabled in the software. To enable IPv6 routing, you must assign IPv6 addresses to individual interfaces in the router using the **ipv6 enable** or **ipv6 address** command. See the Network Stack IPv4 and IPv6 Commands on module of .

Flood Blocking on Specific Interfaces

With this technique, certain interfaces are blocked from being used for flooding LSPs, but the remaining interfaces operate normally for flooding. This technique is simple to understand and configure, but may be more difficult to maintain and more error prone than mesh groups in the long run. The flooding topology that IS-IS uses is fine-tuned rather than restricted. Restricting the topology too much (blocking too many interfaces) makes the network unreliable in the face of failures. Restricting the topology too little (blocking too few interfaces) may fail to achieve the desired scalability.

To improve the robustness of the network in the event that all nonblocked interfaces drop, use the **csnp-interval** command in interface configuration mode to force periodic complete sequence number PDUs (CSNPs) packets to be used on blocked point-to-point links. The use of periodic CSNPs enables the network to become synchronized.

Maximum LSP Lifetime and Refresh Interval

By default, the router sends a periodic LSP refresh every 15 minutes. LSPs remain in a database for 20 minutes by default. If they are not refreshed by that time, they are deleted. You can change the LSP refresh interval or maximum LSP lifetime. The LSP interval should be less than the LSP lifetime or else LSPs time out before they are refreshed. In the absence of a configured refresh interval, the software adjusts the LSP refresh interval, if necessary, to prevent the LSPs from timing out.

Minimum Remaining Lifetime

The Minimum Remaining Lifetime feature prevents premature purging and unnecessary flooding of LSPs. If the Remaining Lifetime field gets corrupted during flooding, this corruption is undetectable. The consequences of such corruption depend on how the Remaining Lifetime value is altered. This feature resolves this problem by enabling IS-IS to reset the Remaining Lifetime value of the received LSP, to the maximum LSP lifetime. By default, the maximum LSP lifetime is configured as 1200 seconds and you can configure it to a different value using the **max-lsp-lifetime** *seconds* command. This action ensures that whatever be the value of

Remaining Lifetime that is received, a system other than the originator of an LSP will never purge the LSP, until the LSP has existed in the database at least for maximum LSP lifetime.

If the remaining lifetime for the LSP reaches 0, the LSP is kept in the link state database for an additional 60 seconds. This additional lifetime is known as Zero Age Lifetime. If the corresponding router does not update the LSP even after the Zero Age Lifetime, the LSP is deleted from the link state database.

The Remaining Lifetime field is also useful in identifying a problem in the network. If the received LSP lifetime value is less than the Zero Age Lifetime, which is 60 seconds, IS-IS generates an error message indicating that it's a corrupted lifetime event. The sample error message is as follows:

```
Dec 14 15:36:45.663 : isis[1011]: RECV L2 LSP 1111.1111.1112.03-00 from 1111.1111.1112.03:
possible corrupted lifetime 59 secs for L2 lsp 1111.1111.1112.03-00 from SNPA 02e9.4522.5326
detected.
```

IS-IS saves the received remaining lifetime value in LSP database. The value is shown in the **show isis database** command output under the **Rcvd** field.

For more information about the **show isis database** command, see *IS-IS Commands* Chapter of the *Routing Command Reference for Cisco NCS 5500 Series Routers*.

Mesh Group Configuration

Configuring mesh groups (a set of interfaces on a router) can help to limit flooding. All routers reachable over the interfaces in a particular mesh group are assumed to be densely connected with each router having at least one link to every other router. Many links can fail without isolating one or more routers from the network.

In normal flooding, a new LSP is received on an interface and is flooded out over all other interfaces on the router. With mesh groups, when a new LSP is received over an interface that is part of a mesh group, the new LSP is not flooded over the other interfaces that are part of that mesh group.

Multitopology IPv6 for IS-IS

Multitopology IPv6 for IS-IS assumes that multitopology support is required as soon as it detects interfaces configured for both IPv6 and IPv4 within the IS-IS stanza.

Because multitopology is the default behavior in the software, you must explicitly configure IPv6 to use the same topology as IPv4 to enable single-topology IPv6. Configure the **single-topology** command in IPv6 router address family configuration submode of the IS-IS router stanza.

The following example shows multitopology IS-IS being configured in IPv6.

```
router isis isp
 net 49.0000.0000.0001.00
 interface POS0/3/0/0
  address-family ipv6 unicast
  metric-style wide level 1
  exit
!
interface POS0/3/0/0
 ipv6 address 2001::1/64
```

IS-IS Authentication

Authentication is available to limit the establishment of adjacencies by using the **hello-password** command, and to limit the exchange of LSPs by using the **lsp-password** command.

IS-IS supports plain-text authentication, which does not provide security against unauthorized users. Plain-text authentication allows you to configure a password to prevent unauthorized networking devices from forming adjacencies with the router. The password is exchanged as plain text and is potentially visible to an agent able to view the IS-IS packets.

When an HMAC-MD5 password is configured, the password is never sent over the network and is instead used to calculate a cryptographic checksum to ensure the integrity of the exchanged data.

IS-IS stores a configured password using simple encryption. However, the plain-text form of the password is used in LSPs, sequence number protocols (SNPs), and hello packets, which would be visible to a process that can view IS-IS packets. The passwords can be entered in plain text (clear) or encrypted form.

To set the domain password, configure the **lsp-password** command for Level 2; to set the area password, configure the **lsp-password** command for Level 1.

The keychain feature allows IS-IS to reference configured keychains. IS-IS key chains enable hello and LSP keychain authentication. Keychains can be configured at the router level (in the case of the **lsp-password** command) and at the interface level (in the case of the **hello-password** command) within IS-IS. These commands reference the global keychain configuration and instruct the IS-IS protocol to obtain security parameters from the global set of configured keychains.

IS-IS is able to use the keychain to implement hitless key rollover for authentication. Key rollover specification is time based, and in the event of clock skew between the peers, the rollover process is impacted. The configurable tolerance specification allows for the accept window to be extended (before and after) by that margin. This accept window facilitates a hitless key rollover for applications (for example, routing and management protocols).

Configure Authentication for IS-IS

This task explains how to configure authentication for IS-IS. This task is optional.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **lsp-password** { **hmac-md5** | **text** } { **clear** | **encrypted** } *password* [**level** { **1** | **2** }] [**send-only**] [**snp send-only**]
4. **interface** *type interface-path-id*
5. **hello-password** { **hmac-md5** | **text** } { **clear** | **encrypted** } *password* [**level** { **1** | **2** }] [**send-only**]
6. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 `router isis instance-id`**Example:**

```
RP/0/RP0/CPU0:router(config)# router isis isp
```

Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.

- You can change the level of routing to be performed by a particular routing instance by using the **is-type** command.

Step 3 `lsp-password { hmac-md5 | text } { clear | encrypted } password [level { 1 | 2 }] [send-only] [snp send-only]`**Example:**

```
RP/0/RP0/CPU0:router(config-isis)# lsp-password hmac-md5 clear password1 level 1
```

Configures the LSP authentication password.

- The **hmac-md5** keyword specifies that the password is used in HMAC-MD5 authentication.
- The **text** keyword specifies that the password uses cleartext password authentication.
- The **clear** keyword specifies that the password is unencrypted when entered.
- The **encrypted** keyword specifies that the password is encrypted using a two-way algorithm when entered.
- The **level 1** keyword sets a password for authentication in the area (in Level 1 LSPs and Level SNPs).
- The **level 2** keywords set a password for authentication in the backbone (the Level 2 area).
- The **send-only** keyword adds authentication to LSP and sequence number protocol data units (SNPs) when they are sent. It does not authenticate received LSPs or SNPs.
- The **snp send-only** keyword adds authentication to SNPs when they are sent. It does not authenticate received SNPs.

Note To disable SNP password checking, the **snp send-only** keywords must be specified in the **lsp-password** command.

Step 4 `interface type interface-path-id`**Example:**

```
RP/0/RP0/CPU0:router(config-isis)# interface GigabitEthernet 0/1/0/3
```

Enters interface configuration mode.

Step 5 `hello-password { hmac-md5 | text } { clear | encrypted } password [level { 1 | 2 }] [send-only]`**Example:**

```
RP/0/RP0/CPU0:router(config-isis-if)#hello-password text clear mypassword
```

Configures the authentication password for an IS-IS interface.

Step 6 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configure Keychains for IS-IS

This task explains how to configure keychains for IS-IS. This task is optional.

Keychains can be configured at the router level (**lsp-password** command) and at the interface level (**hello-password** command) within IS-IS. These commands reference the global keychain configuration and instruct the IS-IS protocol to obtain security parameters from the global set of configured keychains. The router-level configuration (**lsp-password** command) sets the keychain to be used for all IS-IS LSPs generated by this router, as well as for all Sequence Number Protocol Data Units (SN PDUs). The keychain used for HELLO PDUs is set at the interface level, and may be set differently for each interface configured for IS-IS.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **lsp-password keychain** *keychain-name* [**level** { **1** | **2** }] [**send-only**] [**snp send-only**]
4. **interface** *type interface-path-id*
5. **hello-password keychain** *keychain-name* [**level** { **1** | **2** }] [**send-only**]
6. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 **router isis** *instance-id*

Example:

```
RP/0/RP0/CPU0:router(config)# router isis isp
```

Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.

- You can change the level of routing to be performed by a particular routing instance by using the **is-type** command.

Step 3 `lsp-password keychain keychain-name [level { 1 | 2 }] [send-only] [snp send-only]`

Example:

```
RP/0/RP0/CPU0:router(config-isis)# lsp-password keychain isis_a level 1
```

Configures the keychain.

Step 4 `interface type interface-path-id`

Example:

```
RP/0/RP0/CPU0:router(config-isis)# interface HundredGigE 0/1/0/3
```

Enters interface configuration mode.

Step 5 `hello-password keychain keychain-name [level { 1 | 2 }] [send-only]`

Example:

```
RP/0/RP0/CPU0:router(config-isis-if)#hello-password keychain isis_b
```

Configures the authentication password for an IS-IS interface.

Step 6 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Multi-Instance IS-IS

You can configure up to 16 IS-IS instances. MPLS can run on multiple IS-IS processes as long as the processes run on different sets of interfaces. Each interface may be associated with only a single IS-IS instance. The software prevents the double-booking of an interface by two instances at configuration time—two instances of MPLS configuration causes an error.

Because the Routing Information Base (RIB) treats each of the IS-IS instances as equal routing clients, you must be careful when redistributing routes between IS-IS instances. The RIB does not know to prefer Level 1 routes over Level 2 routes. For this reason, if you are running Level 1 and Level 2 instances, you must enforce the preference by configuring different administrative distances for the two instances.

Enable IS-IS and Configure Level 1 or Level 2 Routing

This task explains how to enable IS-IS and configure the routing level for an area.



Note Configuring the routing level in Step 4 is optional, but is highly recommended to establish the proper level of adjacencies.

Before you begin

Although you can configure IS-IS before you configure an IP address, no IS-IS routing occurs until at least one IP address is configured.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **net** *network-entity-title*
4. **is-type** { **level-1** | **level-1-2** | **level-2-only** }
5. Use the **commit** or **end** command.
6. **show isis** [**instance** *instance-id*] **protocol**

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 **router isis** *instance-id*

Example:

```
RP/0/RP0/CPU0:router(config)# router isis isp
```

Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.

- By default, all IS-IS instances are automatically Level 1 and Level 2. You can change the level of routing to be performed by a particular routing instance by using the **is-type** router configuration command.

Step 3 **net** *network-entity-title*

Example:

```
RP/0/RP0/CPU0:router(config-isis)# net 47.0004.004d.0001.0001.0c11.1110.00
```

Configures network entity titles (NETs) for the routing instance.

- Specify a NET for each routing instance if you are configuring multi-instance IS-IS.
- This example configures a router with area ID 47.0004.004d.0001 and system ID 0001.0c11.1110.00.
- To specify more than one area address, specify additional NETs. Although the area address portion of the NET differs, the systemID portion of the NET must match exactly for all of the configured items.

Step 4 **is-type** { **level-1** | **level-1-2** | **level-2-only** }

Example:

```
RP/0/RP0/CPU0:router(config-isis)# is-type level-2-only
```

(Optional) Configures the system type (area or backbone router).

- By default, every IS-IS instance acts as a **level-1-2** router.
- The **level-1** keyword configures the software to perform Level 1 (intra-area) routing only. Only Level 1 adjacencies are established. The software learns about destinations inside its area only. Any packets containing destinations outside the area are sent to the nearest **level-1-2** router in the area.
- The **level-2-only** keyword configures the software to perform Level 2 (backbone) routing only, and the router establishes only Level 2 adjacencies, either with other Level 2-only routers or with **level-1-2** routers.
- The **level-1-2** keyword configures the software to perform both Level 1 and Level 2 routing. Both Level 1 and Level 2 adjacencies are established. The router acts as a border router between the Level 2 backbone and its Level 1 area.

Step 5 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 6 **show isis** [**instance** *instance-id*] **protocol**

Example:

```
RP/0/RP0/CPU0:router# show isis protocol
```

(Optional) Displays summary information about the IS-IS instance.

Single-Topology IPv6

Single-topology IPv6 allows IS-IS for IPv6 to be configured on interfaces along with an IPv4 network protocol. All interfaces must be configured with the identical set of network protocols, and all routers in the IS-IS area (for Level 1 routing) or the domain (for Level 2 routing) must support the identical set of network layer protocols on all interfaces.

In single-topology mode, IPv6 topologies work with both narrow and wide metric styles in IPv4 unicast topology. During single-topology operation, one shortest path first (SPF) computation for each level is used to compute both IPv4 and IPv6 routes. Using a single SPF is possible because both IPv4 IS-IS and IPv6 IS-IS routing protocols share a common link topology.

Configure Single Topology for IS-IS

After an IS-IS instance is enabled, it must be configured to compute routes for a specific network topology.

This task explains how to configure the operation of the IS-IS protocol on an interface for an IPv4 or IPv6 topology.

Before you begin



Note To enable the router to run in single-topology mode, configure each of the IS-IS interfaces with all of the address families enabled and “single-topology” in the address-family IPv6 unicast in the IS-IS router stanza. You can use either the IPv6 address family or both IPv4 and IPv6 address families, but your configuration must represent the set of all active address families on the router. Additionally, explicitly enable single-topology operation by configuring it in the IPv6 router address family submode.

Two exceptions to these instructions exist:

1. If the address-family stanza in the IS-IS process contains the **adjacency-check disable** command, then an interface is not required to have the address family enabled.
2. The **single-topology** command is not valid in the ipv4 address-family submode.

The default metric style for single topology is narrow metrics. However, you can use either wide metrics or narrow metrics. How to configure them depends on how single topology is configured. If both IPv4 and IPv6 are enabled and single topology is configured, the metric style is configured in the **address-family ipv4** stanza. You may configure the metric style in the **address-family ipv6** stanza, but it is ignored in this case. If only IPv6 is enabled and single topology is configured, then the metric style is configured in the **address-family ipv6** stanza.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. Do one of the following:
 - **ipv4 address** *address mask*
 - **ipv6 address** *ipv6-prefix / prefix-length [eui-64]*
 - **ipv6 address** *ipv6-address { / prefix-length | link-local }*
 - **ipv6 enable**
4. **exit**
5. **router isis** *instance-id*
6. **net** *network-entity-title*
7. **address-family ipv6 [unicast]**
8. **single-topology**

9. `exit`
10. `interface type interface-path-id`
11. `circuit-type { level-1 | level-1-2 | level-2-only }`
12. `address-family { ipv4 | ipv6 } [unicast]`
13. Use the `commit` or `end` command.
14. `show isis [instance instance-id] interface [type interface-path-id] [detail] [level { 1 | 2 }]`
15. `show isis [instance instance-id] topology [systemid system-id] [level { 1 | 2 }] [summary]`

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 **interface type interface-path-id**

Example:

```
RP/0/RP0/CPU0:router(config)# interface HundredGigE 0/1/0/3
```

Enters interface configuration mode.

Step 3

Do one of the following:

- **ipv4 address** *address mask*
- **ipv6 address** *ipv6-prefix / prefix-length [eui-64]*
- **ipv6 address** *ipv6-address { / prefix-length | link-local }*
- **ipv6 enable**

Example:

```
RP/0/RP0/CPU0:router(config-if)# ipv4 address 10.0.1.3 255.255.255.0
```

or

```
RP/0/RP0/CPU0:router(config-if)# ipv6 address 3ffe:1234:c18:1::/64 eui-64
RP/0/RP0/CPU0:router(config-if)# ipv6 address FE80::260:3EFF:FE11:6770 link-local
RP/0/RP0/CPU0:router(config-if)# ipv6 enable
```

or

Defines the IPv4 address for the interface. An IP address is required on all interfaces in an area enabled for IS-IS if any one interface is configured for IS-IS routing.

or

Specifies an IPv6 network assigned to the interface and enables IPv6 processing on the interface with the **eui-64** keyword.

or

Specifies an IPv6 address assigned to the interface and enables IPv6 processing on the interface with the **link-local** keyword.

or

Automatically configures an IPv6 link-local address on the interface while also enabling the interface for IPv6 processing.

- The link-local address can be used only to communicate with nodes on the same link.
- Specifying the **ipv6 address** *ipv6-prefix / prefix-length* interface configuration command without the **eui-64** keyword configures site-local and global IPv6 addresses.
- Specifying the **ipv6 address** *ipv6-prefix / prefix-length* command with the **eui-64** keyword configures site-local and global IPv6 addresses with an interface ID in the low-order 64 bits of the IPv6 address. Only the 64-bit network prefix for the address needs to be specified; the last 64 bits are automatically computed from the interface ID.
- Specifying the **ipv6 address** command with the **link-local** keyword configures a link-local address on the interface that is used instead of the link-local address that is automatically configured when IPv6 is enabled on the interface.

Step 4 **exit**

Example:

```
RP/0/RP0/CPU0:router(config-if)# exit
```

Exits interface configuration mode, and returns the router to mode.

Step 5 **router isis** *instance-id*

Example:

```
RP/0/RP0/CPU0:router(config)# router isis isp
```

Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.

- By default, all IS-IS instances are Level 1 and Level 2. You can change the level of routing to be performed by a particular routing instance by using the **is-type** command.

Step 6 **net** *network-entity-title*

Example:

```
RP/0/RP0/CPU0:router(config-isis)# net 47.0004.004d.0001.0001.0c11.1110.00
```

Configures NETs for the routing instance.

- Specify a NET for each routing instance if you are configuring multi-instance IS-IS. You can specify a name for a NET and for an address.
- This example configures a router with area ID 47.0004.004d.0001 and system ID 0001.0c11.1110.00.
- To specify more than one area address, specify additional NETs. Although the area address portion of the NET differs, the system ID portion of the NET must match exactly for all of the configured items.

Step 7 **address-family ipv6** [**unicast**]

Example:

```
RP/0/RP0/CPU0:router(config-isis)# address-family ipv6 unicast
```

Specifies the IPv6 address family and enters router address family configuration mode.

- This example specifies the unicast IPv6 address family.

Step 8 **single-topology**

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# single-topology
```

(Optional) Configures the link topology for IPv4 when IPv6 is configured.

- The **single-topology** command is valid only in IPv6 submode. The command instructs IPv6 to use the single topology rather than the default configuration of a separate topology in the multitopology mode.

Step 9 **exit**

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# exit
```

Exits router address family configuration mode, and returns the router to router configuration mode.

Step 10 **interface** *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router(config-isis)# interface HundredGigE 0/1/0/3HundredGigE 0/1/0/3
```

Enters interface configuration mode.

Step 11 **circuit-type** { **level-1** | **level-1-2** | **level-2-only** }

Example:

```
RP/0/RP0/CPU0:router(config-isis-if)# circuit-type level-1-2
```

(Optional) Configures the type of adjacency.

- The default circuit type is the configured system type (configured through the **is-type** command).
- Typically, the circuit type must be configured when the router is configured as only **level-1-2** and you want to constrain an interface to form only **level-1** or **level-2-only** adjacencies.

Step 12 **address-family** { **ipv4** | **ipv6** } [**unicast**]

Example:

```
RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
```

Specifies the IPv4 or IPv6 address family, and enters interface address family configuration mode.

- This example specifies the unicast IPv4 address family on the interface.

Step 13 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** — Exits the configuration session without committing the configuration changes.
- **Cancel** — Remains in the configuration session, without committing the configuration changes.

Step 14 `show isis [instance instance-id] interface [type interface-path-id] [detail] [level { 1 | 2 }]`

Example:

```
RP/0/RP0/CPU0:router# show isis interface HundredGigE 0/1/0/1
```

(Optional) Displays information about the IS-IS interface.

Step 15 `show isis [instance instance-id] topology [systemid system-id] [level { 1 | 2 }] [summary]`

Example:

```
RP/0/RP0/CPU0:router# show isis topology
```

(Optional) Displays a list of connected routers in all areas.

Configuring Single-Topology IS-IS for IPv6: Example

The following example shows single-topology mode being enabled. An IS-IS instance is created, the NET is defined, IPv6 is configured along with IPv4 on an interface, and IPv4 link topology is used for IPv6. This configuration allows POS interface 0/3/0/0 to form adjacencies for both IPv4 and IPv6 addresses.

```
router isis isp
 net 49.0000.0000.0001.00
 address-family ipv6 unicast
  single-topology
 interface POS0/3/0/0
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  !
  exit
 !
 interface POS0/3/0/0
  ipv4 address 10.0.1.3 255.255.255.0
  ipv6 address 2001::1/64
```

Set SPF Interval for a Single-Topology Configuration

This task explains how to make adjustments to the SPF calculation to tune router performance. This task is optional.

Because the SPF calculation computes routes for a particular topology, the tuning attributes are located in the router address family configuration submenu. SPF calculation computes routes for Level 1 and Level 2 separately.

When IPv4 and IPv6 address families are used in a single-topology mode, only a single SPF for the IPv4 topology exists. The IPv6 topology “borrows” the IPv4 topology; therefore, no SPF calculation is required for IPv6. To tune the SPF calculation parameters for single-topology mode, configure the **address-family ipv4 unicast** command.

The incremental SPF algorithm can be enabled separately. When enabled, the incremental shortest path first (ISPF) is not employed immediately. Instead, the full SPF algorithm is used to “seed” the state information required for the ISPF to run. The startup delay prevents the ISPF from running for a specified interval after an IS-IS restart (to permit the database to stabilize). After the startup delay elapses, the ISPF is principally responsible for performing all of the SPF calculations. The reseed interval enables a periodic running of the full SPF to ensure that the ISPF state remains synchronized.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **address-family** { **ipv4** | **ipv6** } [**unicast**]
4. **spf-interval** {[**initial-wait** *initial* | **secondary-wait** *secondary* | **maximum-wait** *maximum*] ...}
[**level** { **1** | **2** }]
5. **ispf** [**level** { **1** | **2** }]
6. Use the **commit** or **end** command.
7. **show isis** [**instance** *instance-id*] [[**ipv4** | **ipv6** | **afi-all**] [**unicast** | **safi-all**]] **spf-log** [**level** { **1** | **2** }] [**ispf** | **fspf** | **prc** | **nhc**] [**detail** | **verbose**] [**last** *number* | **first** *number*]

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 **router isis** *instance-id*

Example:

```
RP/0/RP0/CPU0:router(config)# router isis isp
```

Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.

- You can change the level of routing to be performed by a particular routing instance by using the **is-type** router configuration command.

Step 3 **address-family** { **ipv4** | **ipv6** } [**unicast**]

Example:

```
RP/0/RP0/CPU0:router(config-isis)#address-family ipv4 unicast
```

Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.

Step 4 `spf-interval` {[**initial-wait** *initial* | **secondary-wait** *secondary* | **maximum-wait** *maximum*] ...} [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# spf-interval initial-wait 10 maximum-wait 30
```

(Optional) Controls the minimum time between successive SPF calculations.

- This value imposes a delay in the SPF computation after an event trigger and enforces a minimum elapsed time between SPF runs.
- If this value is configured too low, the router can lose too many CPU resources when the network is unstable.
- Configuring the value too high delays changes in the network topology that result in lost packets.
- The SPF interval does not apply to the running of the ISPF because that algorithm runs immediately on receiving a changed LSP.

Step 5 `ispf` [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# ispf
```

(Optional) Configures incremental IS-IS ISPF to calculate network topology.

Step 6 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 7 `show isis` [**instance** *instance-id*] [[**ipv4** | **ipv6** | **afi-all**] [**unicast** | **safi-all**]] **spf-log** [**level** { **1** | **2** }] [**ispf** | **fspf** | **prc** | **nhc**] [**detail** | **verbose**] [**last** *number* | **first** *number*]

Example:

```
RP/0/RP0/CPU0:router# show isis instance 1 spf-log ipv4
```

(Optional) Displays how often and why the router has run a full SPF calculation.

Customize Routes for IS-IS

This task explains how to perform route functions that include injecting default routes into your IS-IS routing domain and redistributing routes learned in another IS-IS instance. This task is optional.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **set-overload-bit** [**on-startup** { *delay* | **wait-for-bgp** }] [**level** { **1** | **2** }]
4. **address-family** { **ipv4** | **ipv6** } [**unicast**]
5. **default-information originate** [**route-policy** *route-policy-name*]
6. **redistribute isis** *instance* [**level-1** | **level-2** | **level-1-2**] [**metric** *metric*] [**metric-type** { **internal** | **external** }] [**policy** *policy-name*]
7. Do one of the following:
 - **summary-prefix** *address / prefix-length* [**level** { **1** | **2** }]
 - **summary-prefix** *ipv6-prefix / prefix-length* [**level** { **1** | **2** }]
8. **maximum-paths** *route-number*
9. **distance** *weight* [*address / prefix-length* [*route-list-name*]]
10. **set-attached-bit**
11. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure****Example:**

```
RP/0/# configure
Enters mode.
```

Step 2 **router isis** *instance-id***Example:**

```
RP/0/RP0/CPU0:router(config)# router isis isp
```

Enables IS-IS routing for the specified routing process, and places the router in router configuration mode.

- By default, all IS-IS instances are automatically Level 1 and Level 2. You can change the level of routing to be performed by a particular routing instance by using the **is-type** command.

Step 3 **set-overload-bit** [**on-startup** { *delay* | **wait-for-bgp** }] [**level** { **1** | **2** }]**Example:**

```
RP/0/RP0/CPU0:router(config-isis)# set-overload-bit
```

(Optional) Sets the overload bit.

Note The configured overload bit behavior does not apply to NSF restarts because the NSF restart does not set the overload bit during restart.

Step 4 **address-family** { **ipv4** | **ipv6** } [**unicast**]**Example:**


```
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
```

Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.

Step 5 **default-information originate** [**route-policy** *route-policy-name*]

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# default-information originate
```

(Optional) Injects a default IPv4 or IPv6 route into an IS-IS routing domain.

- The **route-policy** keyword and *route-policy-name* argument specify the conditions under which the IPv4 or IPv6 default route is advertised.
- If the **route-policy** keyword is omitted, then the IPv4 or IPv6 default route is unconditionally advertised at Level 2.

Step 6 **redistribute isis** *instance* [**level-1** | **level-2** | **level-1-2**] [**metric** *metric*] [**metric-type** { **internal** | **external** }] [**policy** *policy-name*]

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# redistribute isis 2 level-1
```

(Optional) Redistributes routes from one IS-IS instance into another instance.

- In this example, an IS-IS instance redistributes Level 1 routes from another IS-IS instance.

Step 7 Do one of the following:

- **summary-prefix** *address / prefix-length* [**level** { **1** | **2** }]
- **summary-prefix** *ipv6-prefix / prefix-length* [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# summary-prefix 10.1.0.0/16 level 1
```

or

```
RP/0/RP0/CPU0:router(config-isis-af)# summary-prefix 3003:xxxx::/24 level 1
```

(Optional) Allows a Level 1-2 router to summarize Level 1 IPv4 and IPv6 prefixes at Level 2, instead of advertising the Level 1 prefixes directly when the router advertises the summary.

- This example specifies an IPv4 address and mask.
- or
- This example specifies an IPv6 prefix, and the command must be in the form documented in RFC 2373 in which the address is specified in hexadecimal using 16-bit values between colons.
 - Note that IPv6 prefixes must be configured only in the IPv6 router address family configuration submode, and IPv4 prefixes in the IPv4 router address family configuration submode.

Step 8 **maximum-paths** *route-number*

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# maximum-paths 16
```

(Optional) Configures the maximum number of parallel paths allowed in a routing table.

Step 9 **distance** *weight* [*address / prefix-length* [*route-list-name*]]

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# distance 90
```

(Optional) Defines the administrative distance assigned to routes discovered by the IS-IS protocol.

- A different administrative distance may be applied for IPv4 and IPv6.

Step 10 **set-attached-bit**

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# set-attached-bit
```

(Optional) Configures an IS-IS instance with an attached bit in the Level 1 LSP.

Step 11 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Redistributing IS-IS Routes Between Multiple Instances: Example

The following example shows usage of the **set- attached-bit** and **redistribute** commands. Two instances, instance “1” restricted to Level 1 and instance “2” restricted to Level 2, are configured.

The Level 1 instance is propagating routes to the Level 2 instance using redistribution. Note that the administrative distance is explicitly configured higher on the Level 2 instance to ensure that Level 1 routes are preferred.

Attached bit is being set for the Level 1 instance since it is redistributing routes into the Level 2 instance. Therefore, instance “1” is a suitable candidate to get from the area to the backbone.

```
router isis 1
  is-type level-2-only
  net 49.0001.0001.0001.0001.00
  address-family ipv4 unicast
  distance 116
  redistribute isis 2 level 2
!
interface HundredGigE 0/3/0/0
  address-family ipv4 unicast
!
```

```

!
router isis 2
 is-type level-1
 net 49.0002.0001.0001.0002.00
 address-family ipv4 unicast
   set
-attached-bit

!
interface HundredGigE 0/1/0/0
 address-family ipv4 unicast

```

Set Priority for Adding Prefixes to RIB

This optional task describes how to set the priority (order) for which specified prefixes are added to the RIB. The prefixes can be chosen using an access list (ACL), prefix list, or by matching a tag value.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **address-family** { **ipv4** | **ipv6** } [**unicast**]
4. **metric-style wide** [**transition**] [**level** { **1** | **2** }]
5. **spf prefix-priority** [**level** { **1** | **2** }] { **critical** | **high** | **medium** } { *access-list-name* | **tag** *tag* }
6. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 **router isis** *instance-id*

Example:

```
RP/0/RP0/CPU0:router(config)# router isis isp
```

Enables IS-IS routing for the specified routing process, and places the router in router configuration mode. In this example, the IS-IS instance is called *isp*.

Step 3 **address-family** { **ipv4** | **ipv6** } [**unicast**]

Example:

```
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
```

Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.

Step 4 `metric-style wide [transition] [level { 1 | 2 }]`

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# metric-style wide level 1
```

Configures a router to generate and accept only wide-link metrics in the Level 1 area.

Step 5 `spf prefix-priority [level { 1 | 2 }] { critical | high | medium } { access-list-name | tag tag }`

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# spf prefix-priority high tag 3
```

Installs all routes tagged with the value 3 first.

Step 6 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

IS-IS Interfaces

IS-IS interfaces can be configured as one of the following types:

- **Active**—advertises connected prefixes and forms adjacencies. This is the default for interfaces.
- **Passive**—advertises connected prefixes but does not form adjacencies. The **passive** command is used to configure interfaces as passive. Passive interfaces should be used sparingly for important prefixes such as loopback addresses that need to be injected into the IS-IS domain. If many connected prefixes need to be advertised then the redistribution of connected routes with the appropriate policy should be used instead.
- **Suppressed**—does not advertise connected prefixes but forms adjacencies. The **suppress** command is used to configure interfaces as suppressed.
- **Shutdown**—does not advertise connected prefixes and does not form adjacencies. The **shutdown** command is used to disable interfaces without removing the IS-IS configuration.

Tag IS-IS Interface Routes

This optional task describes how to associate a tag with a connected route of an IS-IS interface.

SUMMARY STEPS

1. **configure**

2. **router isis** *instance-id*
3. **address-family** { **ipv4** | **ipv6** } [**unicast**]
4. **metric-style wide** [**transition**] [**level** { **1** | **2** }]
5. **exit**
6. **interface** *type number*
7. **address-family** { **ipv4** | **ipv6** } [**unicast**]
8. **tag** *tag*
9. Use the **commit** or **end** command.
10. **show isis** [**ipv4** | **ipv6** | **afi-all**] [**unicast** | **safi-all**] **route** [**detail**]

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
Enters mode.
```

Step 2 **router isis** *instance-id*

Example:

```
RP/0/RP0/CPU0:router(config)# router isis isp
```

Enables IS-IS routing for the specified routing process, and places the router in router configuration mode. In this example, the IS-IS instance is called isp.

Step 3 **address-family** { **ipv4** | **ipv6** } [**unicast**]

Example:

```
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
```

Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.

Step 4 **metric-style wide** [**transition**] [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# metric-style wide level 1
```

Configures a router to generate and accept only wide link metrics in the Level 1 area.

Step 5 **exit**

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# exit
```

Exits router address family configuration mode, and returns the router to router configuration mode.

Step 6 **interface** *type number*

Example:

```
RP/0/RP0/CPU0:router(config-isis)# interface HundredGigE 0/1/0/3
```

Enters interface configuration mode.

Step 7 **address-family** { **ipv4** | **ipv6** } [**unicast**]

Example:

```
RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
```

Specifies the IPv4 or IPv6 address family, and enters address family configuration mode.

Step 8 **tag** *tag*

Example:

```
RP/0/RP0/CPU0:router(config-isis-if-af)# tag 3
```

Sets the value of the tag to associate with the advertised connected route.

Step 9 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 10 **show isis** [**ipv4** | **ipv6** | **afi-all**] [**unicast** | **safi-all**] **route** [**detail**]

Example:

```
RP/0/RP0/CPU0:router(config-isis-if-af)# show isis ipv4 route detail
```

Displays tag information. Verify that all tags are present in the RIB.

Tagging Routes: Example

The following example shows how to tag routes.

```
route-policy isis-tag-55
end-policy
!
route-policy isis-tag-555
  if destination in (5.5.5.0/24 eq 24) then
    set tag 555
    pass
  else
    drop
  endif
end-policy
!
router static
```

```
address-family ipv4 unicast
 0.0.0.0/0 2.6.0.1
 5.5.5.0/24 Null0
!
!
router isis uut
 net 00.0000.0000.12a5.00
 address-family ipv4 unicast
  metric-style wide
  redistribute static level-1 route-policy isis-tag-555
  spf prefix-priority critical tag 13
  spf prefix-priority high tag 444
  spf prefix-priority medium tag 777
```

Nonstop Forwarding

On Cisco IOS XR software, IS-IS NSF minimizes the amount of time a network is unavailable to its users following the restart of the IS-IS process.

When the IS-IS process restarts, all routing peers of that device usually detect that the device went down and then came back up. This transition results in what is called a *routing flap*, which could spread across multiple routing domains. Routing flaps caused by routing restarts create routing instabilities, which are detrimental to the overall network performance. NSF helps to suppress routing flaps, thus reducing network instability.

NSF allows for the forwarding of data packets to continue along known routes while the routing protocol information is being restored following the process restarts. When the NSF feature is configured, peer networking devices do not experience routing flaps. To preserve routing across RP failover events, NSR must be configured in addition to NSF.

When the Cisco IOS XR router running IS-IS routing performs the process restarts, the router must perform two tasks to resynchronize its link-state database with that of its IS-IS neighbors. First, it must relearn the available IS-IS neighbors on the network without causing a reset of the neighbor relationship. Second, it must reacquire the contents of the link-state database for the network.

The IS-IS NSF feature offers two options when configuring NSF:

- IETF NSF
- Cisco NSF

If neighbor routers on a network segment are NSF-aware, meaning that they are running a software version that supports RFC5306, they assist a router configured with **nsf ietf** command that is restarting. IETF NSF enables the neighbor routers provide adjacency and link-state information to help rebuild the routing information following a failover.

In Cisco IOS XR software, Cisco NSF checkpoints (stores persistently) all the state necessary to recover from a restart without requiring any special cooperation from neighboring routers. The state is recovered from the neighboring routers, but only using the standard features of the IS-IS routing protocol. This capability makes Cisco NSF suitable for use in networks in which other routers have not used the IETF standard implementation of NSF.



Note If you configure IETF NSF on the Cisco IOS XR router and a neighbor router does not support IETF NSF, the affected adjacencies flap, but nonstop forwarding is maintained to all neighbors that do support IETF NSF. A restart reverts to a cold start if no neighbors support IETF NSF.

Configure Nonstop Forwarding for IS-IS

This task explains how to configure your router with NSF that allows the software to resynchronize the IS-IS link-state database with its IS-IS neighbors after a process restart. The process restart could be due to an:

- RP failover (for a warm restart)
- Simple process restart (due to an IS-IS reload or other administrative request to restart the process)
- IS-IS software upgrade

In all cases, NSF mitigates link flaps and loss of user sessions. This task is optional.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **nsf** { **cisco** | **ietf** }
4. **nsf interface-expires** *number*
5. **nsf interface-timer** *seconds*
6. **nsf lifetime** *seconds*
7. Use the **commit** or **end** command.
8. **show running-config** [*command*]

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
Enters mode.
```

Step 2 **router isis** *instance-id*

Example:

```
RP/0/RP0/CPU0:router(config)# router isis isp
```

Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.

- You can change the level of routing to be performed by a particular routing instance by using the **is-type** router configuration command.

Step 3 **nsf** { **cisco** | **ietf** }

Example:

```
RP/0/RP0/CPU0:router(config-isis)# nsf ietf
```

Enables NSF on the next restart.

- Enter the **cisco** keyword to run IS-IS in heterogeneous networks that might not have adjacent NSF-aware networking devices.
- Enter the **ietf** keyword to enable IS-IS in homogeneous networks where *all* adjacent networking devices support IETF draft-based restartability.

Step 4 **nsf interface-expires** *number***Example:**

```
RP/0/RP0/CPU0:router(config-isis)# nsf interface-expires 1
```

Configures the number of resends of an acknowledged NSF-restart acknowledgment.

- If the resend limit is reached during the NSF restart, the restart falls back to a cold restart.

Step 5 **nsf interface-timer** *seconds***Example:**

```
RP/0/RP0/CPU0:router(config-isis) nsf interface-timer 15
```

Configures the number of seconds to wait for each restart acknowledgment.

Step 6 **nsf lifetime** *seconds***Example:**

```
RP/0/RP0/CPU0:router(config-isis)# nsf lifetime 20
```

Configures the maximum route lifetime following an NSF restart.

- This command should be configured to the length of time required to perform a full NSF restart because it is the amount of time that the Routing Information Base (RIB) retains the routes during the restart.
- Setting this value too high results in stale routes.
- Setting this value too low could result in routes purged too soon.

Step 7 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 8 **show running-config** [*command*]**Example:**

```
RP/0/RP0/CPU0:router# show running-config router isis isp
```

(Optional) Displays the entire contents of the currently running configuration file or a subset of that file.

- Verify that “nsf” appears in the IS-IS configuration of the NSF-aware device.
- This example shows the contents of the configuration file for the “isp” instance only.

ISIS NSR

Non Stop Routing (NSR) suppresses IS-IS routing changes for devices with redundant route processors during processor switchover events (RP failover or ISSU), reducing network instability and downtime. When Non Stop Routing is used, switching from the active to standby RP have no impact on the other IS-IS routers in the network. All information needed to continue the routing protocol peering state is transferred to the standby processor prior to the switchover, so it can continue immediately upon a switchover.

To preserve routing across process restarts, NSF must be configured in addition to NSR.

Configuring ISIS-NSR

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 **router isis *instance-id***

Example:

```
RP/0/RP0/CPU0:router(config)# router isis 1
```

Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.

Step 3 **nsr**

Example:

```
RP/0/RP0/CPU0:router(config-isis)# nsr
```

Configures the NSR feature.

Step 4 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.

- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 5 **show isis nsr adjacency**

Example:

```
RP/0/RP0/CPU0:router# show isis nsr adjacency
System Id Interface SNPA State Hold Changed NSF IPv4 BFD IPv6 BFD
R1-v1S Nii0 *PtoP* Up 83 00:00:33 Yes None None
```

Displays adjacency information.

Step 6 **show isis nsr status**

Example:

```
RP/0/RP0/CPU0:router# show isis nsr status
IS-IS test NSR(v1a) STATUS (HA Ready):
          V1 Standby V2 Active V2 Standby
SYNC STATUS:          TRUE      FALSE(0)  FALSE(0)
PEER CHG COUNT:      1          0          0
UP TIME:              00:03:12   not up    not up
```

Displays the NSR status information.

Step 7 **show isis nsr statistics**

Example:

```
RP/0/RP0/CPU0:router# show isis nsr statistics
IS-IS test NSR(v1a) MANDATORY STATS :
          V1 Active          V1 Standby          V2 Active          V2
Standby
L1 ADJ:          0          0          0
0
L2 ADJ:          2          2          0
0
LIVE INTERFACE:  4          4          0
0
PTP INTERFACE:   1          1          0
0
LAN INTERFACE:   2          2          0
0
LOOPBACK INTERFACE:  1          1          0
0
TE Tunnel:       1          1          0
0
TE LINK:         2          2          0
0
NSR OPTIONAL STATS :
L1 LSP:          0          0          0
0
L2 LSP:          4          4          0
0
IPV4 ROUTES:     3          3          0
0
IPV6 ROUTES:     4          4          0
0
```

Shows number of ISIS adjacencies, lps, routes, tunnels, Te links on active and standby routers.

Configuring IS-IS Adjacency Stagger

Certain events like process restart or reload can involve a significant processing overhead. Updating routing tables with all adjacencies, maintaining them, and synchronizing the database with each adjacent router requires a lot of bandwidth. These processes may require large number of packets being sent and/or received, depending on the state of the database on the routers. If packets are dropped in any direction, it can lead to an unstable state.

We cannot prevent events like process restart or reload, but we can handle such events better by limiting the number of adjacencies that area being established simultaneously. To limit the number of adjacencies from getting established simultaneously, you can configure adjacency stagger. By configuring IS-IS adjacency stagger, you can specify the initial number neighbourhood routers from which adjacencies can fully form after a process restart or reload. If you configure IS-IS adjacency stagger, you can also specify the subsequent number of simultaneous neighbors that are allowed to form adjacency.

Restrictions

- IS-IS adjacency stagger is only supported on point-to-point interfaces and not on LAN interfaces.
- IS-IS adjacency stagger is not supported with NSF (non-stop forwarding) mechanisms.

Configuration Example

To configure IS-IS adjacency stagger on a point-to-point interface, you must use the following configuration steps:

1. Configure IS-IS.
2. Configure adjacency stagger.

Configuration

```
/* Enter the global configuration mode and configure IS-IS */
Router# config
Router(config)# router isis 1

/* Configure IS-IS adjacency stagger */
Router(config-isis)# adjacency stagger 2 3
Router(config-isis)# commit
```

Multiprotocol Label Switching Traffic Engineering

The MPLS TE feature enables an MPLS backbone to replicate and expand the traffic engineering capabilities of Layer 2 ATM and Frame Relay networks. MPLS is an integration of Layer 2 and Layer 3 technologies.

For IS-IS, MPLS TE automatically establishes and maintains MPLS TE label-switched paths across the backbone by using Resource Reservation Protocol (RSVP). The route that a label-switched path uses is determined by the label-switched paths resource requirements and network resources, such as bandwidth.

Available resources are flooded by using special IS-IS TLV extensions in the IS-IS. The label-switched paths are explicit routes and are referred to as traffic engineering (TE) tunnels.

Configure MPLS Traffic Engineering for IS-IS

This task explains how to configure IS-IS for MPLS TE. This task is optional.

Before you begin

Your network must support the MPLS software feature before you enable MPLS TE for IS-IS on your router.



Note You must enter the commands in the following task list on every IS-IS router in the traffic-engineered portion of your network.



Note MPLS traffic engineering currently does not support routing and signaling of LSPs over unnumbered IP links. Therefore, do not configure the feature over those links.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **address-family** { **ipv4** | **ipv6** } [**unicast**]
4. **mpls traffic-eng level** { **1** | **2** }
5. **mpls traffic-eng router-id** { *ip-address* | *interface-name interface-instance* }
6. **metric-style wide** [**level** { **1** | **2** }]
7. Use the **commit** or **end** command.
8. **show isis** [**instance** *instance-id*] **mpls traffic-eng tunnel**
9. **show isis** [**instance** *instance-id*] **mpls traffic-eng adjacency-log**
10. **show isis** [**instance** *instance-id*] **mpls traffic-eng advertisements**

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
Enters mode.
```

Step 2 **router isis** *instance-id*

Example:

```
RP/0/RP0/CPU0:router(config)# router isis isp
```

Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.

- You can change the level of routing to be performed by a particular routing instance by using the **is-type** router configuration command.

Step 3 **address-family** { **ipv4** | **ipv6** } [**unicast**]

Example:

```
RP/0/RP0/CPU0:router(config-isis)#address-family ipv4 unicast
```

Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.

Step 4 **mpls traffic-eng level** { **1** | **2** }

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# mpls traffic-eng level 1
```

Configures a router running IS-IS to flood MPLS TE link information into the indicated IS-IS level.

Step 5 **mpls traffic-eng router-id** { *ip-address* | *interface-name interface-instance* }

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# mpls traffic-eng router-id loopback0
```

Specifies that the MPLS TE router identifier for the node is the given IP address or an IP address associated with the given interface.

Step 6 **metric-style wide** [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis-af)# metric-style wide level 1
```

Configures a router to generate and accept only wide link metrics in the Level 1 area.

Step 7 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 8 **show isis** [**instance** *instance-id*] **mpls traffic-eng tunnel**

Example:

```
RP/0/RP0/CPU0:router# show isis instance isp mpls traffic-eng tunnel
```

(Optional) Displays MPLS TE tunnel information.

Step 9 **show isis** [**instance** *instance-id*] **mpls traffic-eng adjacency-log**

Example:

```
RP/0/RP0/CPU0:router# show isis instance isp mpls traffic-eng adjacency-log
```

(Optional) Displays a log of MPLS TE IS-IS adjacency changes.

Step 10 `show isis [instance instance-id] mpls traffic-eng advertisements`

Example:

```
RP/0/RP0/CPU0:router# show isis instance isp mpls traffic-eng advertisements
```

(Optional) Displays the latest flooded record from MPLS TE.

MPLS TE Forwarding Adjacency

MPLS TE forwarding adjacency allows a network administrator to handle a traffic engineering, label switch path (LSP) tunnel as a link in an Interior Gateway Protocol (IGP) network, based on the Shortest Path First (SPF) algorithm. A forwarding adjacency can be created between routers in the same IS-IS level. The routers can be located multiple hops from each other. As a result, a TE tunnel is advertised as a link in an IGP network, with the cost of the link associated with it. Routers outside of the TE domain see the TE tunnel and use it to compute the shortest path for routing traffic throughout the network.

MPLS TE forwarding adjacency is considered in IS-IS SPF only if a two-way connectivity check is achieved. This is possible if the forwarding adjacency is bidirectional or the head end and tail end routers of the MPLS TE tunnel are adjacent.

The MPLS TE forwarding adjacency feature is supported by IS-IS. For details on configuring MPLS TE forwarding adjacency, see the MPLS Configuration Guide.

Tune Adjacencies for IS-IS

This task explains how to enable logging of adjacency state changes, alter the timers for IS-IS adjacency packets, and display various aspects of adjacency state. Tuning your IS-IS adjacencies increases network stability when links are congested. This task is optional.

For point-to-point links, IS-IS sends only a single hello for Level 1 and Level 2, which means that the level modifiers are meaningless on point-to-point links. To modify hello parameters for a point-to-point interface, omit the specification of the level options.

The options configurable in the interface submode apply only to that interface. By default, the values are applied to both Level 1 and Level 2.

The **hello-password** command can be used to prevent adjacency formation with unauthorized or undesired routers. This ability is particularly useful on a LAN, where connections to routers with which you have no desire to establish adjacencies are commonly found.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **log adjacency changes**
4. **interface** *type interface-path-id*
5. **hello-padding** { **disable** | **sometimes** } [**level** { **1** | **2** }]

6. **hello-interval** *seconds* [**level** { **1** | **2** }]
7. **hello-multiplier** *multiplier* [**level** { **1** | **2** }]
8. **hello-password** { **hmac-md5** | **text** } { **clear** | **encrypted** } *password* [**level** { **1** | **2** }] [**send-only**]
9. Use the **commit** or **end** command.
10. **show isis** [**instance** *instance-id*] **adjacency** *type interface-path-id* [**detail**] [**systemid** *system-id*]
11. **show isis adjacency-log**
12. **show isis** [**instance** *instance-id*] **interface** [*type interface-path-id*] [**brief** | **detail**] [**level** { **1** | **2** }]
13. **show isis** [**instance** *instance-id*] **neighbors** [*interface-type interface-instance*] [**summary**] [**detail**] [**systemid** *system-id*]

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 **router isis** *instance-id*

Example:

```
RP/0/RP0/CPU0:router(config)# router isis isp
```

Enables IS-IS routing for the specified routing instance, and places the router in router configuration mode.

- You can change the level of routing to be performed by a particular routing instance by using the **is-type** command.

Step 3 **log adjacency changes**

Example:

```
RP/0/RP0/CPU0:router(config-isis)# log adjacency changes
```

Generates a log message when an IS-IS adjacency changes state (up or down).

Step 4 **interface** *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router(config-isis)# interface HundredGigE 0/1/0/3
```

Enters interface configuration mode.

Step 5 **hello-padding** { **disable** | **sometimes** } [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis-if)# hello-padding sometimes
```


Configures padding on IS-IS hello PDUs for an IS-IS interface on the router.

- Hello padding applies to only this interface and not to all interfaces.

Step 6 **hello-interval** *seconds* [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis-if)#hello-interval 6
```

Specifies the length of time between hello packets that the software sends.

Step 7 **hello-multiplier** *multiplier* [**level** { **1** | **2** }]

Example:

```
RP/0/RP0/CPU0:router(config-isis-if)# hello-multiplier 10
```

Specifies the number of IS-IS hello packets a neighbor must miss before the router should declare the adjacency as down.

- A higher value increases the networks tolerance for dropped packets, but also may increase the amount of time required to detect the failure of an adjacent router.
- Conversely, not detecting the failure of an adjacent router can result in greater packet loss.

Step 8 **hello-password** { **hmac-md5** | **text** } { **clear** | **encrypted** } *password* [**level** { **1** | **2** }] [**send-only**]

Example:

```
RP/0/RP0/CPU0:router(config-isis-if)# hello-password text clear mypassword
```

Specifies that this system include authentication in the hello packets and requires successful authentication of the hello packet from the neighbor to establish an adjacency.

Step 9 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 10 **show isis** [**instance** *instance-id*] **adjacency** *type interface-path-id* [**detail**] [**systemid** *system-id*]

Example:

```
RP/0/RP0/CPU0:router# show isis instance isp adjacency
```

(Optional) Displays IS-IS adjacencies.

Step 11 **show isis adjacency-log**

Example:

```
RP/0/RP0/CPU0:router# show isis adjacency-log
```

(Optional) Displays a log of the most recent adjacency state transitions.

Step 12 `show isis [instance instance-id] interface [type interface-path-id] [brief | detail] [level { 1 | 2 }]`

Example:

```
RP/0/RP0/CPU0:router# show isis interface HundredGigE 0/1/0/1 brief
```

(Optional) Displays information about the IS-IS interface.

Step 13 `show isis [instance instance-id] neighbors [interface-type interface-instance] [summary] [detail] [systemid system-id]`

Example:

```
RP/0/RP0/CPU0:router# show isis neighbors summary
```

(Optional) Displays information about IS-IS neighbors.

MPLS Label Distribution Protocol IGP Synchronization

Multiprotocol Label Switching (MPLS) Label Distribution Protocol (LDP) Interior Gateway Protocol (IGP) Synchronization ensures that LDP has completed label exchange before the IGP path is used for switching. MPLS traffic loss can occur in the following two situations:

- When an IGP adjacency is established, the router begins forwarding packets using the new adjacency before LDP has exchanged labels with peers on that link.
- When an LDP session closes, the router continues to forward traffic using the link associated with the LDP peer rather than using an alternate path with an established LDP session.

This feature provides a mechanism to synchronize LDP and IS-IS to minimize MPLS packet loss. The synchronization is accomplished by changing the link metric for a neighbor IS-IS link-state packet (LSP), based on the state of the LDP session.

When an IS-IS adjacency is established on a link but the LDP session is lost or LDP has not yet completed exchanging labels, IS-IS advertises the maximum metric on that link. In this instance, LDP IS-IS synchronization is not yet achieved.



Note In IS-IS, a link with a maximum wide metric (0xFFFFFFFF) is not considered for shortest path first (SPF). Therefore, the maximum wide metric of -1 (0xFFFFFFFFE) is used with MPLS LDP IGP synchronization.

When LDP IS-IS synchronization is achieved, IS-IS advertises a regular (configured or default) metric on that link.

Configuring MPLS LDP IS-IS Synchronization

This task explains how to enable Multiprotocol Label Switching (MPLS) Label Distribution Protocol (LDP) IS-IS synchronization. MPLS LDP synchronization can be enabled for an address family under interface configuration mode. Only IPv4 unicast address family is supported. This task is optional.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **interface** *type interface-path-id*
4. **address-family ipv4 unicast**
5. **mpls ldp sync** [**level** { **1** | **2** }]
6. Use the **commit** or **end** command.

DETAILED STEPS**Step 1** **configure****Example:**

```
RP/0/# configure
Enters mode.
```

Step 2 **router isis** *instance-id***Example:**

```
RP/0/RP0/CPU0:router(config)# router isis isp
```

Enables IS-IS routing for the specified routing process, and places the router in router configuration mode.

- By default, all IS-IS instances are automatically Level 1 and Level 2. You can change the level of routing to be performed by a particular routing instance by using the **is-type** command.

Step 3 **interface** *type interface-path-id***Example:**

```
RP/0/RP0/CPU0:router(config-isis)# interface HundredGigE 0/1/0/3
Enters interface configuration mode.
```

Step 4 **address-family ipv4 unicast****Example:**

```
RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
```

Specifies the IPv4 address family and enters router address family configuration mode.

Step 5 **mpls ldp sync** [**level** { **1** | **2** }]**Example:**

```
RP/0/RP0/CPU0:router(config-isis-if-af)# mpls ldp sync level 1
```

Enables MPLS LDP synchronization for the IPv4 address family under interface HundredGigE 0/1/0/3.

Step 6 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end—Prompts user to take one of these actions:

- **Yes**—Saves configuration changes and exits the configuration session.
- **No**—Exits the configuration session without committing the configuration changes.
- **Cancel**—Remains in the configuration session, without committing the configuration changes.

Overload Bit on Router

The overload bit is a special bit of state information that is included in an LSP of the router. If the bit is set on the router, it notifies routers in the area that the router is not available for transit traffic. This capability is useful in four situations:

1. During a serious but nonfatal error, such as limited memory.
2. During the startup and restart of the process. The overload bit can be set until the routing protocol has converged. However, it is not employed during a normal NSF restart or failover because doing so causes a routing flap.
3. During a trial deployment of a new router. The overload bit can be set until deployment is verified, then cleared.
4. During the shutdown of a router. The overload bit can be set to remove the router from the topology before the router is removed from service.

Overload Bit Configuration During Multitopology Operation

Because the overload bit applies to forwarding for a single topology, it may be configured and cleared independently for IPv4 and IPv6 during multitopology operation. For this reason, the overload is set from the router address family configuration mode. If the IPv4 overload bit is set, all routers in the area do not use the router for IPv4 transit traffic. However, they can still use the router for IPv6 transit traffic.

IS-IS Overload Bit Avoidance

The IS-IS overload bit avoidance feature allows network administrators to prevent label switched paths (LSPs) from being disabled when a router in that path has its Intermediate System-to-Intermediate System (IS-IS) overload bit set.

When the IS-IS overload bit avoidance feature is activated, all nodes with the overload bit set, including head nodes, mid nodes, and tail nodes, are ignored, which means that they are still available for use with label switched paths (LSPs).



Note The IS-IS overload bit avoidance feature does *not* change the default behavior on nodes that have their overload bit set if those nodes are not included in the path calculation (PCALC).

The IS-IS overload bit avoidance feature is activated using the following command:

```
mpls traffic-eng path-selection ignore overload
```

The IS-IS overload bit avoidance feature is deactivated using the **no** form of this command:

```
no mpls traffic-eng path-selection ignore overload
```

When the IS-IS overload bit avoidance feature is deactivated, nodes with the overload bit set cannot be used as nodes of last resort.

Configure IS-IS Overload Bit Avoidance

This task describes how to activate IS-IS overload bit avoidance.

Before you begin

The IS-IS overload bit avoidance feature is valid only on networks that support the following features:

- MPLS
- IS-IS

SUMMARY STEPS

1. **configure**
2. **mpls traffic-eng path-selection ignore overload**

DETAILED STEPS

Step 1 **configure****Example:**

```
RP/0/# configure
```

Enters mode.

Step 2 **mpls traffic-eng path-selection ignore overload****Example:**

```
RP/0/RP0/CPU0:router(config)# mpls traffic-eng path-selection ignore overload
```

Activates IS-IS overload bit avoidance.

Configuring IS-IS Overload Bit Avoidance: Example

The following example shows how to activate IS-IS overload bit avoidance:

```
config
```

```
mpls traffic-eng path-selection ignore overload
```

The following example shows how to deactivate IS-IS overload bit avoidance:

```
config
no mpls traffic-eng path-selection ignore overload
```

Default Routes

You can force a default route into an IS-IS routing domain. Whenever you specifically configure redistribution of routes into an IS-IS routing domain, the software does not, by default, redistribute the default route into the IS-IS routing domain. The **default-information originate** command generates a *default route* into IS-IS, which can be controlled by a route policy. You can use the route policy to identify the level into which the default route is to be announced, and you can specify other filtering options configurable under a route policy. You can use a route policy to conditionally advertise the default route, depending on the existence of another route in the routing table of the router.

Attached Bit on an IS-IS Instance

The attached bit is set in a router that is configured with the **is-type** command and **level-1-2** keyword. The attached bit indicates that the router is connected to other areas (typically through the backbone). This functionality means that the router can be used by Level 1 routers in the area as the default route to the backbone. The attached bit is usually set automatically as the router discovers other areas while computing its Level 2 SPF route. The bit is automatically cleared when the router becomes detached from the backbone.



Note If the connectivity for the Level 2 instance is lost, the attached bit in the Level 1 instance LSP would continue sending traffic to the Level 2 instance and cause the traffic to be dropped.

To simulate this behavior when using multiple processes to represent the **level-1-2** keyword functionality, you would manually configure the attached bit on the Level 1 process.

IS-IS Support for Route Tags

The IS-IS Support for route tags feature provides the capability to associate and advertise a tag with an IS-IS route prefix. Additionally, the feature allows you to prioritize the order of installation of route prefixes in the RIB based on a tag of a route. Route tags may also be used in route policy to match route prefixes (for example, to select certain route prefixes for redistribution).

Multicast-Intact Feature

The multicast-intact feature provides the ability to run multicast routing (PIM) when IGP shortcuts are configured and active on the router. Both OSPFv2 and IS-IS support the multicast-intact feature. MPLS TE

and IP multicast coexistence is supported in Cisco IOS XR software by using the **mpls traffic-eng multicast-intact** IS-IS or OSPF router command.

You can enable multicast-intact in the IGP when multicast routing protocols (PIM) are configured and IGP shortcuts are configured on the router. IGP shortcuts are MPLS tunnels that are exposed to IGP. The IGP's route the IP traffic over these tunnels to destinations that are downstream from the egress router of the tunnel (from an SPF perspective). PIM cannot use IGP shortcuts for propagating PIM joins because reverse path forwarding (RPF) cannot work across a unidirectional tunnel.

When you enable multicast-intact on an IGP, the IGP publishes a parallel or alternate set of equal-cost next-hops for use by PIM. These next-hops are called mcast-intact next-hops. The mcast-intact next-hops have the following attributes:

- They are guaranteed not to contain any IGP shortcuts.
- They are not used for unicast routing but are used only by PIM to look up an IPv4 next-hop to a PIM source.
- They are not published to the FIB.
- When multicast-intact is enabled on an IGP, all IPv4 destinations that were learned through link-state advertisements are published with a set equal-cost mcast-intact next-hops to the RIB. This attribute applies even when the native next-hops have no IGP shortcuts.
- In IS-IS, the max-paths limit is applied by counting both the native and mcast-intact next-hops together. (In OSPFv2, the behavior is slightly different.)

Multicast Topology Support Using IS-IS

Multicast topology support allows for the configuration of IS-IS multicast topologies for IPv4 or IPv6 routing. IS-IS maintains a separate topology for multicast and runs a separate Shortest Path First (SPF) over the multicast topology. IS-IS multicast inserts routes from the IS-IS multicast topology into the multicast-unicast Routing Information Base (muRIB) table in the RIB for the corresponding address family. Since PIM uses the muRIB, PIM uses routes from the multicast topology instead of routes from the unicast topology.

MPLS TE Interarea Tunnels

MPLS TE interarea tunnels allow you to establish MPLS TE tunnels that span multiple IGP areas (Open Shorted Path First [OSPF]) and levels (IS-IS), removing the restriction that required that both the tunnel headend and tailend routers be in the same area. The IGP can be either IS-IS or OSPF.

For details on configuring MPLS TE interarea tunnels, see the MPLS Configuration Guide.

IP Fast Reroute

The IP Fast Reroute (IPFRR) loop-free alternate (LFA) computation provides protection against link failure. Locally computed repair paths are used to prevent packet loss caused by loops that occur during network reconvergence after a failure. See IETF draft-ietf-rtgwg-ipfrr-framework-06.txt and draft-ietf-rtgwg-lf-conv-frmwk-00.txt for detailed information on IPFRR LFA.

IPFRR LFA is different from Multiprotocol Label Switching (MPLS) as it is applicable to networks using conventional IP routing and forwarding. See for information on configuring MPLS IPFRR.

Unequal Cost Multipath Load-balancing for IS-IS

The unequal cost multipath (UCMP) load-balancing adds the capability with intermediate system-to-intermediate system (IS-IS) to load-balance traffic proportionally across multiple paths, with different cost.

Generally, higher bandwidth links have lower IGP metrics configured, so that they form the shortest IGP paths. With the UCMP load-balancing enabled, IGP can use even lower bandwidth links or higher cost links for traffic, and can install these paths to the forwarding information base (FIB). IS-IS IGP still installs multiple paths to the same destination in FIB, but each path will have a 'load metric/weight' associated with it. FIB uses this load metric/weight to decide the amount of traffic that needs to be sent on a higher bandwidth path and the amount of traffic that needs to be sent on a lower bandwidth path.

The UCMP computation is provided under IS-IS per address family, enabling UCMP computation for a particular address family. The UCMP configuration is also provided with a prefix-list option, which would limit the UCMP computation only for the prefixes present in the prefix-list. If prefix-list option is not provided, UCMP computation is done for the reachable prefixes in IS-IS. The number of UCMP nexthops to be considered and installed is controlled using the **variance** configuration. Variance value identifies the range for the UCMP path metric to be considered for installation into routing information base (RIB) and is defined in terms of a percentage of the primary path metric. Total number of paths, including ECMP and UCMP paths together is limited by the max-path configuration or by the max-path capability of the platform.

Enabling the UCMP configuration indicates that IS-IS should perform UCMP computation for the all the reachable ISIS prefixes or all the prefixes in the prefix-list, if the prefix-list option is used. The UCMP computation happens only after the primary SPF and route calculation is completed. There would be a delay of `ISIS_UCMP_INITIAL_DELAY` (default delay is 100 ms) milliseconds from the time route calculation is completed and UCMP computation is started. UCMP computation will be done before fast re-route computation. Fast re-route backup paths will be calculated for both the primary equal cost multipath (ECMP) paths and the UCMP paths. Use the **ucmp delay-interval** command to configure the delay between primary SPF completion and start of UCMP computation.

UCMP ratio can be adjusted by any of the following ways:

- By using the **bandwidth** command in interface configuration mode .
- By adjusting ISIS metric on the links.

There is an option to exclude an interface from being used for UCMP computation. If it is desired that a particular interface should not be considered as a UCMP nexthop, for any prefix, then use the **ucmp exclude interface** command to configure the interface to be excluded from UCMP computation.

Configuring Multitopology Routing

This set of procedures configures multitopology routing, which is used by PIM for reverse-path forwarding (RPF) path selection.

Restrictions for Configuring Multitopology Routing

- Only protocol-independent multicast (PIM) and intermediate system-intermediate system (IS-IS) routing protocols are currently supported.
- Topology selection is restricted solely to (S, G) route sources for both SM and SSM. Static and IS-IS are the only interior gateway protocols (IGPs) that support multitopology deployment.

For non-(S, G) route sources like a rendezvous point or bootstrap router (BSR), or when a route policy is not configured, the current policy default remains in effect. In other words, either a unicast-default or multicast-default table is selected for all sources, based on OSPF/IS-IS/Multiprotocol Border Gateway Protocol (MBGP) configuration.

Information About Multitopology Routing

Configuring multitopology networks requires the following tasks:

Configuring a Global Topology and Associating It with an Interface

Follow these steps to enable a global topology in the default VRF and to enable its use with a specific interface.

SUMMARY STEPS

1. **configure**
2. **address-family** { **ipv4** | **ipv6** } **multicast topology** *topo-name*
3. **maximum prefix** *limit*
4. **interface** *type interface-path-id*
5. **address-family** { **ipv4** | **ipv6** } **multicast topology** *topo-name*
6. Repeat Step 4 and Step 5 until you have specified all the interface instances you want to associate with your topologies.
7. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	address-family { ipv4 ipv6 } multicast topology <i>topo-name</i> Example:	Configures a topology in the default VRF table that will be associated with a an interface.

	Command or Action	Purpose
	Router(config)# address-family ipv4 multicast topology green	
Step 3	maximum prefix <i>limit</i> Example: Router(config-af)# maximum prefix 100	(Optional) Limits the number of prefixes allowed in a topology routing table. Range is 32 to 2000000.
Step 4	interface <i>type interface-path-id</i> Example: Router(config-af)# interface GigabitEthernet 0/3/0/0	Specifies the interface to be associated with the previously specified VRF table that will add the connected and local routes to the appropriate routing table.
Step 5	address-family { ipv4 ipv6 } multicast topology <i>topo-name</i> Example: Router(config-if)# address-family ipv4 multicast topology green	Enables the topology for the interface specified in Step 4, adding the connected and local routes to the appropriate routing table.
Step 6	Repeat Step 4 and Step 5 until you have specified all the interface instances you want to associate with your topologies. Example: Router(config-if-af)# interface gigabitethernet 0/3/2/0 Router(config-if)# address-family ipv4 multicast topology purple Router(config-if-af)#	—
Step 7	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Enabling an IS-IS Topology

To enable a topology in IS-IS, you must associate an IS-IS topology ID with the named topology. IS-IS uses the topology ID to differentiate topologies in the domain.



Note This command must be configured prior to other topology commands.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **address-family** { **ipv4** | **ipv6** } **multicast topology** *topo-name*
4. **topology-id** *multitopology-id*
5. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	router isis <i>instance-id</i> Example: RP/0/(config)# router isis purple	Enters IS-IS configuration submode.
Step 3	address-family { ipv4 ipv6 } multicast topology <i>topo-name</i> Example: RP/0/(config-isis)# address-family ipv4 multicast topology green	Associates an IS-IS topology ID with the named topology.
Step 4	topology-id <i>multitopology-id</i> Example: RP/0/(config-isis-af)# topology-id 122	Configures the numeric multitopologyID in IS-IS that identifies the topology. Range is 6 to 4095.
Step 5	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session.

	Command or Action	Purpose
		<ul style="list-style-type: none"> • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Placing an Interface in a Topology in IS-IS

To associate an interface with a topology in IS-IS, follow these steps.

Step 1 **configure**

Example:

```
RP/0/# configure
Enters mode.
```

Step 2 **router isis** *instance-id*

Example:

```
Routing(config)# router isis purple
Enters IS-IS configuration submode.
```

Step 3 **net** *network-entity-title*

Example:

```
Routing(config-isis)# net netname
Creates a network entity title for the configured isis interface.
```

Step 4 **interface** *type interface-path-id*

Example:

```
Routing(config-isis)# interface gigabitethernet 0/3/0/0
Enters isis interface configuration submode and creates an interface instance.
```

Step 5 **address-family** { **ipv4** | **ipv6** } **multicast topology** *topo-name*

Example:

```
Routing(config-isis-if)# address-family ipv4 multicast topology green
```

- Enters isis address-family interface configuration submode.
- Places the interface instance into a topology.

Step 6 Repeat Step 4 and Step 5 until you have specified all the interface instances and associated topologies you want to configure in your network.

Step 7 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring a Routing Policy

For more information about creating a routing policy and about the **set rpf-topology** command, see .

SUMMARY STEPS

1. **configure**
2. **route-policy** *policy-name*
3. **end-policy**
4. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	route-policy <i>policy-name</i> Example: RP/0/(config)# route-policy mt1 RP/0/(config-rpl)# if destination in 225.0.0.1, 225.0.0.11 then RP/0/(config-rpl-if)# if source in (10.10.10.10) then RP/0/(config-rpl-if-2)# set rpf-topology ipv4 multicast topology greentable RP/0/(config-rpl-if-2)# else RP/0/(config-rpl-if-else-2)# set rpf-topology ipv4 multicast topology bluetable RP/0/(config-rpl-if-else-2)# endif RP/0/(config-rpl-if)# endif	Defines a routing policy and enters routing policy configuration submenu. For detailed information about the use of the set-rpf-topology and other routing configuration commands, see .

	Command or Action	Purpose
Step 3	end-policy Example: RP/0/(config-rpl)# end-policy RP/0/(config)#	Signifies the end of route policy definition and exits routing policy configuration submode.
Step 4	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Configuring Multitopology for IS-IS

Multitopology is configured in the same way as the single topology. However, the **single - topology** command is omitted, invoking the default multitopology behavior. This task is optional.

Enabling Multicast-Intact

This optional task describes how to enable multicast-intact for IS-IS routes that use IPv4 and IPv6 addresses.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **address-family** { **ipv4** | **ipv6** } [**unicast** | **multicast**]
4. **mpls traffic-eng multicast-intact**
5. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.

	Command or Action	Purpose
Step 2	router isis <i>instance-id</i> Example: RP/0/(config)# router isis isp	Enables IS-IS routing for the specified routing process, and places the router in router configuration mode. In this example, the IS-IS instance is called isp.
Step 3	address-family { ipv4 ipv6 } [unicast multicast] Example: RP/0/(config-isis)# address-family ipv4 unicast	Specifies the IPv4 or IPv6 address family, and enters router address family configuration mode.
Step 4	mpls traffic-eng multicast-intact Example: RP/0/(config-isis-af)# mpls traffic-eng multicast-intact	Enables multicast-intact.
Step 5	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Configuring IP/LDP Fast Reroute

This optional task describes how to enable the IP/LDP fast reroute computation to converge traffic flows around link failures.



Note To enable node protection on broadcast links, fast reroute and bidirectional forwarding detection (BFD) must be enabled on the interface under IS-IS.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **interface** *type interface-path-id*
4. **circuit-type** { **level-1** | **level-1-2** | **level-2-only** }
5. **address-family** { **ipv4** | **ipv6** } [**unicast**]

6. **fast-reroute** {per-link | per-prefix}
7. Do one of the following:
 - **fast-reroute per-link** { level { 1 | 2 }}
 - **fast-reroute per-prefix** { level { 1 | 2 }}
8. Do one of the following:
 - **fast-reroute per-link exclude interface** *type interface-path-id* { level { 1 | 2 }}
 - **fast-reroute per-prefix exclude interface** *type interface-path-id* { level { 1 | 2 }}
9. Do one of the following:
 - **fast-reroute per-link lfa-candidate interface** *type interface-path-id* { level { 1 | 2 }}
 - **fast-reroute per-prefix lfa-candidate interface** *type interface-path-id* { level { 1 | 2 }}
10. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	router isis <i>instance-id</i> Example: RP/0/(config)# router isis isp	Enables IS-IS routing for the specified routing process, and places the router in router configuration mode. In this example, the IS-IS instance is called isp.
Step 3	interface <i>type interface-path-id</i> Example: RP/0/(config-isis)# interface GigabitEthernet 0/1/0/3	Enters interface configuration mode.
Step 4	circuit-type { level-1 level-1-2 level-2-only } Example: RP/0/(config-isis-if)# circuit-type level-1	(Optional) Configures the type of adjacency.
Step 5	address-family { ipv4 ipv6 } [unicast] Example: RP/0/(config-isis-if)# address-family ipv4 unicast	Specifies the address family, and enters router address family configuration mode. <ul style="list-style-type: none"> • This example specifies the unicast IPv4 address family.

	Command or Action	Purpose
Step 6	<p>fast-reroute {per-link per-prefix}</p> <p>Example:</p> <pre>RP/0/8(config-isis-if-af)# fast-reroute per-link</pre>	<p>Specifies fast-reroute computation on per-link or per-prefix basis.</p> <ul style="list-style-type: none"> • per-link—Used for prefix independent per-link computation. • per-prefix—Used for prefix dependent computation.
Step 7	<p>Do one of the following:</p> <ul style="list-style-type: none"> • fast-reroute per-link { level { 1 2 }} • fast-reroute per-prefix { level { 1 2 }} <p>Example:</p> <pre>RP/0/ (config-isis-if-af)#fast-reroute per-link level 1</pre> <p>Or</p> <pre>RP/0/ (config-isis-if-af)#fast-reroute per-prefix level 2</pre>	<p>Configures fast-reroute per-link or per-prefix computation for one level; use either level 1 or level 2.</p>
Step 8	<p>Do one of the following:</p> <ul style="list-style-type: none"> • fast-reroute per-link exclude interface <i>type interface-path-id</i> { level { 1 2 }} • fast-reroute per-prefix exclude interface <i>type interface-path-id</i> { level { 1 2 }} <p>Example:</p> <pre>RP/0/ (config-isis-if-af)#fast-reroute per-link exclude interface Loopback0 level 1</pre> <p>Or</p> <pre>RP/0/ (config-isis-if-af)#fast-reroute per-prefix exclude interface POS0/6/0/0 level 2</pre>	<p>Excludes an interface from fast-reroute computation.</p>
Step 9	<p>Do one of the following:</p> <ul style="list-style-type: none"> • fast-reroute per-link lfa-candidate interface <i>type interface-path-id</i> { level { 1 2 }} • fast-reroute per-prefix lfa-candidate interface <i>type interface-path-id</i> { level { 1 2 }} <p>Example:</p> <pre>RP/0/ (config-isis-if-af)#fast-reroute per-link lfa-candidate interface MgmtEth0/RP0/CPU0/0 level 1</pre> <p>Or</p> <pre>RP/0/ (config-isis-if-af)#fast-reroute per-prefix lfa-candidate interface MgmtEth0/RP1/CPU0/0 level 2</pre>	<p>Configures to include an interface to LFA candidate in fast-reroute computation.</p>
Step 10	<p>Use the commit or end command.</p>	<p>commit —Saves the configuration changes and remains within the configuration session.</p>

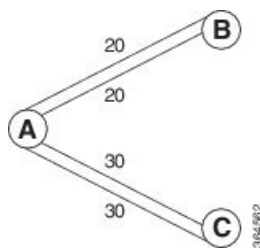
	Command or Action	Purpose
		<p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

ISIS Link Group

The ISIS Link-Group feature allows you to define a group or set of links, and raise or lower their ISIS metric according to a predefined number of active links.

When the total number of active links (in terms of ISIS adjacency) in a group falls below the configured number or members, a predefined offset is applied on the remaining active links. When the total number of active links in a group is reverted, ISIS restores the configured metric by removing the offset.

In the example below, Router A has to exit through router B and C. In between A and B there are two layer 3 links with the same ISIS metric (20). There is a similar setup between A and C (30). In normal operations, the traffic from A goes through B. If the ISIS Link-Group is not configured, even when the link between A and B fails, traffic is still routed through B. However, with ISIS Link-Group, you can set an offset of 20 with minimum-members of 2. Thus, if a link between A and B fails, the metric is raised to 40 (configured (20) + offset (20)), and so the traffic is routed to C. Further, you can define another ISIS Link-Group, this time between A and C. If a link between B and C fails, you can raise the offset to 20, and thus traffic is routed back to B.



Configure Link Group Profile

Perform this task to configure Intermediate System-to-Intermediate System (IS-IS) link group profiles:

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **link-group** *link-group-name* { [**metric-offset** *count* | **maximum**] | [**minimum-members** *count* | **revert-members** *count*] }

4. Use the **commit** or **end** command.
5. **show isis interface**
6. **show isis lsp**

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	router isis instance-id Example: RP/0/(config)# router isis purple	Enters IS-IS configuration submode.
Step 3	link-group link-group-name { [metric-offset count maximum] [minimum-members count revert-members count] }	<p>Specifies link-group values. Following are the valid values:</p> <ul style="list-style-type: none"> • metric-offset: Configures the metric offset for link group. The range is 1-16777214. The default metric offset range is between 1-63 for narrow metric; and 1-16777214 for wide metric. <p>The maximum option here sets the maximum wide metric offset. All routers exclude this link from their SPF.</p> <ul style="list-style-type: none"> • minimum-members: Configures the minimum number of members in the link group. The range is 2-64. • revert-members: Configures the number of members after which to revert in the link group. The range is 2-64. <p>Note A link-group is only active after the minimum-members and offset-metric are configured in the profile. The revert-members is default to minimum-members if it is not configured.</p>
Step 4	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes.

	Command or Action	Purpose
		<ul style="list-style-type: none"> • Cancel —Remains in the configuration session, without committing the configuration changes.
Step 5	show isis interface Example: RP/0/# show isis interface	(Optional) If link-group is configured on the interface, when showing the IS-IS interface-related topology, this command displays the link-group and current offset-metric value.
Step 6	show isis lsp Example: RP/0/# show isis lsp	(Optional) Displays the updated metric value.

Configure Link Group Profile: Example

The following is an example configuration, along with the show isis interface output:

```

router isis 1
 is-type level-2-only
 net 49.1111.0000.0000.0006.00
 link-group foo
  metric-offset 100
  revert-members 4
  minimum-members 2
 !
 address-family ipv4 unicast
  metric-style wide
 !
 interface GigabitEthernet0/0/0/1
  point-to-point
  address-family ipv4 unicast
   link-group foo

RP/0/RSP0/CPU0:Iguazu#sh isis interface gig 0/0/0/1
Thu Jun 11 14:55:32.565 CEST

GigabitEthernet0/0/0/1      Enabled
Adjacency Formation:      Enabled
Prefix Advertisement:     Enabled
IPv4 BFD:                 Disabled
IPv6 BFD:                 Disabled
BFD Min Interval:        150
BFD Multiplier:           3

Circuit Type:              level-2-only (Interface circuit type is level-1-2)
Media Type:                P2P
Circuit Number:           0
Extended Circuit Number:   36
Next P2P IIH in:          8 s
LSP Rermit Queue Size:    0

Level-2
Adjacency Count:          1
LSP Pacing Interval:      33 ms
PSNP Entry Queue Size:    0

```

```

CLNS I/O
  Protocol State:      Up
  MTU:                1497
  SNPA:               0026.9829.af19
  Layer-2 MCast Groups Membership:
    All ISs:          Yes

IPv4 Unicast Topology: Enabled
  Adjacency Formation: Running
  Prefix Advertisement: Running
  Metric (L1/L2):     110/110
  Weight (L1/L2):     0/0
  MPLS Max Label Stack: 1
  MPLS LDP Sync (L1/L2): Disabled/Disabled
  Link-Group (L1/L2): Configured/Configured
  Metric-Offset (L1/L2): 100/100

IPv4 Address Family: Enabled
  Protocol State:      Up
  Forwarding Address(es): 100.5.6.6
  Global Prefix(es):   100.5.6.0/24

LSP transmit timer expires in 0 ms
LSP transmission is idle
Can send up to 9 back-to-back LSPs in the next 0 ms

```

Configure Link Group Interface

Perform this task to configure link group under Intermediate System-to-Intermediate System (IS-IS) interface and address-family sub-mode:



Note One IS-IS interface and address-family can specify only one link-group association. The default is for both levels regardless of the current circuit-type. The link-group association can be specified for one level only if configured.

SUMMARY STEPS

1. **configure**
2. **router isis** *instance-id*
3. **interface** *type interface-path-id*
4. **address-family** **ipv4** | **ipv6** [**unicast**]
5. **link-group** *link-group-name* [**level** { **1** | **2** }]
6. Use the **commit** or **end** command.
7. **show isis interface**

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example:	Enters mode.

	Command or Action	Purpose
	RP/0/# configure	
Step 2	router isis <i>instance-id</i> Example: RP/0/(config)# router isis purple	Enters IS-IS configuration submode.
Step 3	interface <i>type interface-path-id</i> Example: RP/0/(config-isis)# interface GigabitEthernet 0/1/0/3	Enters interface configuration mode.
Step 4	address-family ipv4 ipv6 [unicast] Example: RP/0/(config-isis)# address-family ipv4 unicast	Specifies the IPv6 address family and enters router address family configuration mode. <ul style="list-style-type: none">• This example specifies the unicast IPv4 address family.
Step 5	link-group <i>link-group-name</i> [level { 1 2 }] Example: RP/0/(config-isis-if)#)#address-family ipv4 unicast link-group access level 1	Specifies the link-group name and sets the tag at the level specified.
Step 6	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none">• Yes — Saves configuration changes and exits the configuration session.• No —Exits the configuration session without committing the configuration changes.• Cancel —Remains in the configuration session, without committing the configuration changes.
Step 7	show isis interface Example: RP/0/# show isis interface	(Optional) If link-group is configured on the interface, when showing the IS-IS interface-related topology, this command displays the link-group value.

Configuration Examples for Implementing IS-IS

This section provides the following configuration examples:

Configuring Single-Topology IS-IS for IPv6: Example

The following example shows single-topology mode being enabled. An IS-IS instance is created, the NET is defined, IPv6 is configured along with IPv4 on an interface, and IPv4 link topology is used for IPv6.

This configuration allows POS interface 0/3/0/0 to form adjacencies for both IPv4 and IPv6 addresses.

```
router isis isp
 net 49.0000.0000.0001.00
 address-family ipv6 unicast
  single-topology
 interface POS0/3/0/0
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  !
 exit
!
interface POS0/3/0/0
 ipv4 address 10.0.1.3 255.255.255.0
 ipv6 address 2001::1/64
```

Configuring Multitopology IS-IS for IPv6: Example

The following example shows multitopology IS-IS being configured in IPv6.

```
router isis isp
 net 49.0000.0000.0001.00
 interface POS0/3/0/0
  address-family ipv6 unicast
  metric-style wide level 1
 exit
!
interface POS0/3/0/0
 ipv6 address 2001::1/64
```

Redistributing IS-IS Routes Between Multiple Instances: Example

The following example shows usage of the **attached-bit** and **redistribute** commands. Two instances, instance “1” restricted to Level 1 and instance “2” restricted to Level 2, are configured.

The Level 1 instance is propagating routes to the Level 2 instance using redistribution. Note that the administrative distance is explicitly configured higher on the Level 2 instance to ensure that Level 1 routes are preferred.

Attached bit is being set for the Level 1 instance since it is redistributing routes into the Level 2 instance. Therefore, instance “1” is a suitable candidate to get from the area to the backbone.

```
router isis 1
 is-type level-2-only
 net 49.0001.0001.0001.0001.00
 address-family ipv4 unicast
 distance 116
 redistribute isis 2 level 2
!
```

```

interface GigabitEthernet 0/3/0/0
 address-family ipv4 unicast
 !
 !
router isis 2
 is-type level-1
 net 49.0002.0001.0001.0002.00
 address-family ipv4 unicast
-
-

!
interface GigabitEthernet 0/1/0/0
 address-family ipv4 unicast

```

Tagging Routes: Example

The following example shows how to tag routes.

```

route-policy isis-tag-55
end-policy
!
route-policy isis-tag-555
 if destination in (5.5.5.0/24 eq 24) then
   set tag 555
   pass
 else
   drop
 endif
end-policy
!
router static
 address-family ipv4 unicast
  0.0.0.0/0 2.6.0.1
  5.5.5.0/24 Null0
 !
!
router isis uut
 net 00.0000.0000.12a5.00
 address-family ipv4 unicast
 metric-style wide
 redistribute static level-1 route-policy isis-tag-555
 spf prefix-priority critical tag 13
 spf prefix-priority high tag 444
 spf prefix-priority medium tag 777

```

Configuring IS-IS Overload Bit Avoidance: Example

The following example shows how to activate IS-IS overload bit avoidance:

```

config
 mpls traffic-eng path-selection ignore overload

```

The following example shows how to deactivate IS-IS overload bit avoidance:


```

config
no mpls traffic-eng path-selection ignore overload

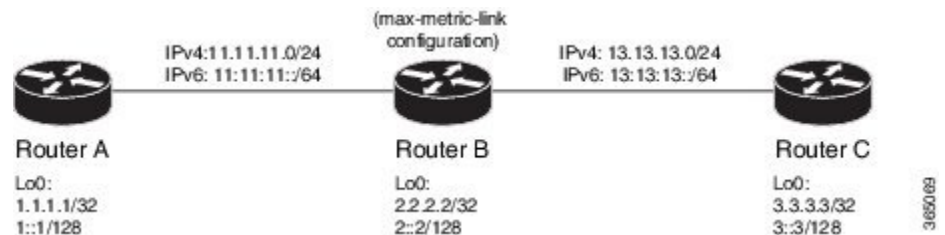
```

Example: Configuring IS-IS To Handle Router Overload

This section describes an example for configuring IS-IS to handle overloading of routers, without setting the overload bit.

When a router is configured with the IS-IS overload bit, it participates in the routing process when the overload bit is set, but does not forward traffic (except for traffic to directly connected interfaces). To configure the overload behavior for IS-IS, without setting the overload bit, configure the **max-metric** statement. By configuring this statement, the router participates in the routing process and is used as a transit node of last resort.

Figure 1:



Before you begin

Ensure that you are familiar with configuring router interfaces for a given topology.

SUMMARY STEPS

1. Configure Routers A, B, and C as shown in the topology.
2. Configure IS-IS and the corresponding net addresses on Routers A, B and C.
3. Configure IPv4 and IPv6 address families on the loopback interfaces of Routers A, B, and C.
4. Configure the link metrics on the router interfaces.
5. Confirm your configuration by viewing the route prefixes on Routers A, B, and C.
6. Confirm the link metrics on Router B, prior to configuring the **max-metric** statement.
7. Configure the **max-metric** statement on Router B.
8. Commit your configuration.
9. Confirm the change in link metrics on Router B.
10. (Optional) Verify the change in route prefixes on Routers A and C.

DETAILED STEPS

Step 1 Configure Routers A, B, and C as shown in the topology.

Use the following IP Addresses:

- **Router A Loopback0:** 1.1.1.1/32 and 1::1/128
- **Router A -> Router B:** 11.11.11.2/24 and 11:11:11::2/64

- **Router B Loopback0:** 2.2.2.2/32 and 2::2/128
- **Router B -> Router A:** 11.11.11.1/24 and 11:11:11::1/64
- **Router B-> Router C:** 13.13.13.1/24 and 13:13:13::1/64
- **Router C Loopback0:** 3.3.3.3/32 and 3::3/128
- **Router C-> Router B:** 13.13.13.2/24 and 13:13:13::2/64

Step 2 Configure IS-IS and the corresponding net addresses on Routers A, B and C.

Example:

```
!Router A
RP/0/0/CPU0:RouterA(config)# router isis ring
RP/0/0/CPU0:RouterA(config-isis)# net 00.0000.0000.0001.00
RP/0/0/CPU0:RouterA(config-isis)# address-family ipv4 unicast
RP/0/0/CPU0:RouterA(config-isis)# metric-style wide
RP/0/0/CPU0:RouterA(config-isis-af)# exit

!Router B
RP/0/0/CPU0:RouterB(config)# router isis ring
RP/0/0/CPU0:RouterB(config-isis)# net 00.0000.0000.0002.00
RP/0/0/CPU0:RouterB(config-isis)# address-family ipv4 unicast
RP/0/0/CPU0:RouterB(config-isis-af)# exit

!Router C
RP/0/0/CPU0:RouterC(config)# router isis ring
RP/0/0/CPU0:RouterC(config-isis)# net 00.0000.0000.0003.00
RP/0/0/CPU0:RouterC(config-isis)# address-family ipv4 unicast
RP/0/0/CPU0:RouterC(config-isis)# metric-style wide
RP/0/0/CPU0:RouterC(config-isis-af)# exit
```

Step 3 Configure IPv4 and IPv6 address families on the loopback interfaces of Routers A, B, and C.

Example:

```
RP/0/0/CPU0:Router(config-isis)# interface loopback0
RP/0/0/CPU0:Router(config-isis-if)# address-family ipv4 unicast
RP/0/0/CPU0:Router(config-isis-if-af)# exit
RP/0/0/CPU0:Router(config-isis-if)# address-family ipv6 unicast
RP/0/0/CPU0:Router(config-isis-if-af)# exit
RP/0/0/CPU0:Router(config-isis-if)# exit
RP/0/0/CPU0:Router(config-isis)#
```

Step 4 Configure the link metrics on the router interfaces.

Example:

```
! Configuration for Router A Interface GigabitEthernet 0/0/0/0 with Router B is shown here. Similarly,
  configure other router interfaces.
RP/0/0/CPU0:RouterA(config-isis)# interface GigabitEthernet 0/0/0/0
RP/0/0/CPU0:RouterA(config-isis-if)# address-family ipv4 unicast
RP/0/0/CPU0:RouterA(config-isis-if-af)# metric 10
RP/0/0/CPU0:RouterA(config-isis-if-af)# exit
RP/0/0/CPU0:RouterA(config-isis-if)# address-family ipv6 unicast
RP/0/0/CPU0:RouterA(config-isis-if-af)# exit
RP/0/0/CPU0:RouterA(config-isis-if)# exit
RP/0/0/CPU0:RouterA(config-isis)#
```

Step 5 Confirm your configuration by viewing the route prefixes on Routers A, B, and C.

Example:

```
! The outputs for Router A are shown here. Similarly, view the outputs for Routers B and C.
RP/0/0/CPU0:RouterA# show route
Tue Oct 13 13:55:18.342 PST
```

```
Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
U - per-user static route, o - ODR, L - local, G - DAGR, l - LISIP
A - access/subscriber, a - Application route
M - mobile route, (!) - FRR Backup path
```

```
Gateway of last resort is not set
```

```
L 1.1.1.1/32 is directly connected, 00:03:40, Loopback0
i L1 2.2.2.2/32 [115/20] via 11.11.11.2, 00:01:27, GigabitEthernet0/0/0/0
i L1 3.3.3.3/32 [115/30] via 11.11.11.2, 00:01:27, GigabitEthernet0/0/0/0
C 11.11.11.0/24 is directly connected, 00:03:39, GigabitEthernet0/0/0/0
L 11.11.11.1/32 is directly connected, 00:03:39, GigabitEthernet0/0/0/0
i L1 13.13.13.0/24 [115/20] via 11.11.11.2, 00:01:27, GigabitEthernet0/0/0/0
i L1 15.15.15.0/24 [115/30] via 11.11.11.2, 00:01:27, GigabitEthernet0/0/0/0
```

```
RP/0/0/CPU0:RouterA# show route ipv6
Tue Oct 13 14:00:55.758 PST
```

```
Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
U - per-user static route, o - ODR, L - local, G - DAGR, l - LISIP
A - access/subscriber, a - Application route
M - mobile route, (!) - FRR Backup path
```

```
Gateway of last resort is not set
```

```
L 1::1/128 is directly connected,
  00:09:17, Loopback0
i L1 2::2/128
  [115/20] via fe80::e9:45ff:fe22:5326, 00:00:05, GigabitEthernet0/0/0/0
i L1 3::3/128
  [115/30] via fe80::e9:45ff:fe22:5326, 00:00:05, GigabitEthernet0/0/0/0
C 11:11:11::/64 is directly connected,
  00:09:16, GigabitEthernet0/0/0/0
L 11:11:11::1/128 is directly connected,
  00:09:16, GigabitEthernet0/0/0/0
i L1 13:13:13::/64
  [115/20] via fe80::e9:45ff:fe22:5326, 00:00:05, GigabitEthernet0/0/0/0
i L1 15:15:15::/64
  [115/30] via fe80::e9:45ff:fe22:5326, 00:00:05, GigabitEthernet0/0/0/0
```

Step 6

Confirm the link metrics on Router B, prior to configuring the **max-metric** statement.

Example:

```
RP/0/0/CPU0:RouterB# show isis database
Tue Oct 13 13:56:44.077 PST
```

```
No IS-IS RING levels found
```

```

IS-IS ring (Level-1) Link State Database
LSPID          LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
RouterB.00-00  * 0x00000005  0x160d        1026          0/0/0
  Area Address: 00
  NLPID:        0xcc
  NLPID:        0x8e
  MT:           Standard (IPv4 Unicast)
  MT:           IPv6 Unicast
  Hostname:     RouterB
  IP Address:   2.2.2.2
  IPv6 Address: 2::2

  Metric: 10    IS RouterB.01
  Metric: 10    IS RouterA.00
  Metric: 10    IP 2.2.2.2/32
  Metric: 10    IP 11.11.11.0/24
  Metric: 10    IP 13.13.13.0/24
  Metric: 10    MT (IPv6 Unicast) IS-Extended RouterB.01
  Metric: 10    MT (IPv6 Unicast) IS-Extended RouterA.00
  Metric: 10    MT (IPv6 Unicast) IPv6 2::2/128
  Metric: 10    MT (IPv6 Unicast) IPv6 11:11:11::/64
  Metric: 10    MT (IPv6 Unicast) IPv6 13:13:13::/64
RouterB.01-00  0x00000001  0xc8df        913          0/0/0
  Metric: 0     IS RouterB.00
  Metric: 0     IS RouterC.00
  Metric: 0     IS-Extended RouterB.00
  Metric: 0     IS-Extended RouterC.00

Total Level-1 LSP count: 2      Local Level-1 LSP count: 1

```

The output verifies that IS-IS protocol is operational and the displayed link metrics (**Metric: 10**) are as configured.

Step 7 Configure the **max-metric** statement on Router B.

Example:

```

RP/0/0/CPU0:RouterB(config)# router isis ring
RP/0/0/CPU0:RouterB(config-isis)# max-metric
RP/0/0/CPU0:RouterB(config-isis)# exit
RP/0/0/CPU0:RouterB(config)#

```

Step 8 Commit your configuration.

Example:

```

RP/0/0/CPU0:RouterB(config)# commit

```

Step 9 Confirm the change in link metrics on Router B.

Example:

```

RP/0/0/CPU0:RouterB# show isis database
Tue Oct 13 13:58:36.790 PST

No IS-IS RING levels found
IS-IS ring (Level-1) Link State Database
LSPID          LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
RouterB.00-00  * 0x00000006  0x0847        1171          0/0/0
  Area Address: 00
  NLPID:        0xcc
  NLPID:        0x8e
  MT:           Standard (IPv4 Unicast)
  MT:           IPv6 Unicast
  Hostname:     RouterB
  IP Address:   2.2.2.2
  IPv6 Address: 2::2

```

```

Metric: 63          IS RouterB.01
Metric: 63          IS RouterA.00
Metric: 63          IP 2.2.2.2/32
Metric: 63          IP 11.11.11.0/24
Metric: 63          IP 13.13.13.0/24
Metric: 16777214   MT (IPv6 Unicast) IS-Extended RouterB.01
Metric: 16777214   MT (IPv6 Unicast) IS-Extended RouterA.00
Metric: 16777214   MT (IPv6 Unicast) IPv6 2::2/128
Metric: 16777214   MT (IPv6 Unicast) IPv6 11:11:11::/64
Metric: 16777214   MT (IPv6 Unicast) IPv6 13:13:13::/64
RouterB.01-00      0x00000001  0xc8df      800          0/0/0
Metric: 0          IS RouterB.00
Metric: 0          IS RouterC.00
Metric: 0          IS-Extended RouterB.00
Metric: 0          IS-Extended RouterC.00

```

```
Total Level-1 LSP count: 2      Local Level-1 LSP count: 1
```

The output verifies that maximum link metrics (**63** for IPv4 and **16777214** for IPv6) have been allocated for the designated links.

Step 10

(Optional) Verify the change in route prefixes on Routers A and C.

Example:

```
! The outputs for Router A are shown here. Similarly, view the outputs on Router C.
RP/0/0/CPU0:RouterA# show route
Tue Oct 13 13:58:59.289 PST
```

```
Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
U - per-user static route, o - ODR, L - local, G - DAGR, l - LISP
A - access/subscriber, a - Application route
M - mobile route, (!) - FRR Backup path
```

```
Gateway of last resort is not set
```

```
L 1.1.1.1/32 is directly connected, 00:07:21, Loopback0
i L1 2.2.2.2/32 [115/73] via 11.11.11.2, 00:00:50, GigabitEthernet0/0/0/0
i L1 3.3.3.3/32 [115/83] via 11.11.11.2, 00:00:50, GigabitEthernet0/0/0/0
C 11.11.11.0/24 is directly connected, 00:07:20, GigabitEthernet0/0/0/0
L 11.11.11.1/32 is directly connected, 00:07:20, GigabitEthernet0/0/0/0
i L1 13.13.13.0/24 [115/73] via 11.11.11.2, 00:00:50, GigabitEthernet0/0/0/0
i L1 15.15.15.0/24 [115/83] via 11.11.11.2, 00:00:50, GigabitEthernet0/0/0/0
```

```
RP/0/0/CPU0:RouterA# show route ipv6
Tue Oct 13 14:00:06.616 PST
```

```
Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
U - per-user static route, o - ODR, L - local, G - DAGR, l - LISP
A - access/subscriber, a - Application route
M - mobile route, (!) - FRR Backup path
```

```
Gateway of last resort is not set
```

```

L 1::1/128 is directly connected,
  00:08:28, Loopback0
i L1 2::2/128
  [115/16777224] via fe80::e9:45ff:fe22:5326, 00:01:58, GigabitEthernet0/0/0/0
i L1 3::3/128
  [115/16777234] via fe80::e9:45ff:fe22:5326, 00:01:58, GigabitEthernet0/0/0/0
C 11:11:11::/64 is directly connected,
  00:08:27, GigabitEthernet0/0/0/0
L 11:11:11::1/128 is directly connected,
  00:08:27, GigabitEthernet0/0/0/0
i L1 13:13:13::/64
  [115/16777224] via fe80::e9:45ff:fe22:5326, 00:01:58, GigabitEthernet0/0/0/0
i L1 15:15:15::/64
  [115/16777234] via fe80::e9:45ff:fe22:5326, 00:01:58, GigabitEthernet0/0/0/0

```

The output verifies the impact of maximum metric configuration in the routing table: [115/73] and [115/83]

IS-IS has been successfully configured to handle router overload without setting the overload bit.

Configuring Global Weighted SRLG Protection

A shared risk link group (SRLG) is a set of links sharing a common resource and thus shares the same risk of failure. The existing loop-free alternate (LFA) implementations in interior gateway protocols (IGPs) support SRLG protection. However, the existing implementation considers only the directly connected links while computing the backup path. Hence, SRLG protection may fail if a link that is not directly connected but shares the same SRLG is included while computing the backup path. Global weighted SRLG protection feature provides better path selection for the SRLG by associating a weight with the SRLG value and using the weights of the SRLG values while computing the backup path.

To support global weighted SRLG protection, you need information about SRLGs on all links in the area topology. You can flood SRLGs for remote links using ISIS or manually configuring SRLGS on remote links.

Configuration Examples: Global Weighted SRLG Protection

There are three types of configurations that are supported for the global weighted SRLG protection feature.

- local SRLG with global weighted SRLG protection
- remote SRLG flooding
- remote SRLG static provisioning

This example shows how to configure the local SRLG with global weighted SRLG protection feature.

```

RP/0/RP0/CPU0:router(config)# srlg
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/1
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg)# name group value 100
RP/0/RP0/CPU0:router(config)# router isis 1
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix srlg-protection
weighted-global
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix tiebreaker srlg-disjoint

```

```

index 1
RP/0/RP0/CPU0:router(config-isis)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-isis-if)# point-to-point
RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
RP/0/RP0/CPU0:router(config-isis)# srlg
RP/0/RP0/CPU0:router(config-isis-srlg)# name group1
RP/0/RP0/CPU0:router(config-isis-srlg-name)# admin-weight 5000

```

This example shows how to configure the global weighted SRLG protection feature with remote SRLG flooding. The configuration includes local and remote router configuration. On the local router, the global weighted SRLG protection is enabled by using the **fast-reroute per-prefix srlg-protection weighted-global** command. In the remote router configuration, you can control the SRLG value flooding by using the **advertise application lfa link-attributes srlg** command. You should also globally configure SRLG on the remote router.

The local router configuration for global weighted SRLG protection with remote SRLG flooding is as follows:

```

RP/0/RP0/CPU0:router(config)# router isis 1
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix srlg-protection
weighted-global
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix tiebreaker srlg-disjoint
index 1
RP/0/RP0/CPU0:router(config-isis-if-af)# exit
RP/0/RP0/CPU0:router(config-isis)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-isis-if)# point-to-point
RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
RP/0/RP0/CPU0:router(config-isis-if-af)# exit
RP/0/RP0/CPU0:router(config-isis)# srlg
RP/0/RP0/CPU0:router(config-isis-srlg)# name group1
RP/0/RP0/CPU0:router(config-isis-srlg-name)# admin-weight 5000

```

The remote router configuration for global weighted SRLG protection with remote SRLG flooding is as follows:

```

RP/0/RP0/CPU0:router(config)# srlg
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/1
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg)# name group value 100
RP/0/RP0/CPU0:router(config-srlg)# exit
RP/0/RP0/CPU0:router(config)# router isis 1
RP/0/RP0/CPU0:(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-af)# advertise application lfa link-attributes srlg

```

This example shows configuring the global weighted SRLG protection feature with static provisioning of SRLG values for remote links. You should perform these configurations on the local router.

```

RP/0/RP0/CPU0:router(config)# srlg
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg-if)# exit
RP/0/RP0/CPU0:router(config-srlg)# interface TenGigE0/0/0/1
RP/0/RP0/CPU0:router(config-srlg-if)# name group1
RP/0/RP0/CPU0:router(config-srlg)# name group value 100

```

```

RP/0/RP0/CPU0:router(config-srlg)# exit
RP/0/RP0/CPU0:router(config)# router isis 1
RP/0/RP0/CPU0:router(config-isis)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix srlg-protection
weighted-global
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix tiebreaker srlg-disjoint
index 1
RP/0/RP0/CPU0:router(config-isis)# interface TenGigE0/0/0/0
RP/0/RP0/CPU0:router(config-isis-if)# point-to-point
RP/0/RP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix
RP/0/RP0/CPU0:router(config-isis-if-af)# fast-reroute per-prefix ti-lfa
RP/0/RP0/CPU0:router(config-isis)# srlg
RP/0/RP0/CPU0:router(config-isis-srlg)# name group1
RP/0/RP0/CPU0:router(config-isis-srlg-name)# admin-weight 5000
RP/0/RP0/CPU0:router(config-isis-srlg-name)# static ipv4 address 10.0.4.1 next-hop ipv4
address 10.0.4.2
RP/0/RP0/CPU0:router(config-isis-srlg-name)# static ipv4 address 10.0.4.2 next-hop ipv4
address 10.0.4.1

```

Label Distribution Protocol IGP Auto-configuration

Label Distribution Protocol (LDP) Interior Gateway Protocol (IGP) auto-configuration simplifies the procedure to enable LDP on a set of interfaces used by an IGP instance. LDP IGP auto-configuration can be used on a large number of interfaces (for example, when LDP is used for transport in the core) and on multiple IGP instances simultaneously.

This feature supports the IPv4 address family for the default VPN routing and forwarding (VRF) instance.

LDP IGP auto-configuration can also be explicitly disabled on individual interfaces under LDP using the **igp auto-config disable** command. This allows LDP to receive all IGP interfaces except the ones explicitly disabled.

See the MPLS configuration guide for information on configuring LDP IGP auto-configuration.

MPLS LDP-IGP Synchronization Compatibility with LDP Graceful Restart

LDP graceful restart protects traffic when an LDP session is lost. If a graceful restart-enabled LDP session fails, MPLS LDP IS-IS synchronization is still achieved on the interface while it is protected by graceful restart. MPLS LDP IGP synchronization is eventually lost under the following circumstances:

- LDP fails to restart before the LDP graceful restart reconnect timer expires.
- The LDP session on the protected interface fails to recover before the LDP graceful restart recovery timer expires.

MPLS LDP-IGP Synchronization Compatibility with IGP Nonstop Forwarding

IS-IS nonstop forwarding (NSF) protects traffic during IS-IS process restarts and route processor (RP) failovers. LDP IS-IS synchronization is supported with IS-IS NSF only if LDP graceful restart is also enabled over the interface. If IS-IS NSF is not enabled, the LDP synchronization state is not retained across restarts and failovers.

Management Information Base (MIB) for IS-IS

Cisco IOS XR supports full MIBs and traps for IS-IS, as defined in RFC 4444. The RFC 4444 defines objects of the Management Information Base (MIB) for use with the IS-IS Routing Protocol.

To know more about MIBS, please use the [MIB Locator](#).

Support for a Configurable Knob to Reject ISIS PDU on L2 Interfaces

Table 2: Feature History Table

Feature Name	Release Information	Feature Description
Support for a Configurable Knob to Reject ISIS PDU on Layer 2 Interfaces	Release 7.3.1	<p>This feature enables you to use Layer 2 ACL to drop ISIS packets from certain ISIS destination MAC addresses. Dropping ISIS packets allows you to isolate a particular node from ISIS domain. This feature enables you to utilize the network bandwidth efficiently.</p> <p>This feature introduces the ethernet-services access-list isis-drop-all-l2-pdus command</p>

If you bind Layer 2 ACL to Layer 2 physical main interface or subinterface, or bundle main interface or subinterface, the ISIS TRAP is enabled at the main interface ethernet port. These interfaces share the same ethernet port. So even if you configure Layer 2 ACL either on the main interface or subinterface, the ISIS traffic reaching both the main and subinterfaces is dropped. The main or sub interfaces can be Layer 3 or Layer 2. You can configure only one Layer 2 ACL configuration which you can apply to multiple interfaces.

Restrictions

- This feature is supported only in the ingress direction.
- Per-interface statistics is not supported.
- Layer 2 ACL modification is not supported.
- Only remarks can be added, updated or modified.
- Any insertion or modification of Layer ACL access control entries (ACE) is rejected. However, deletion of ACE is accepted as it cannot be blocked.
- If you delete the ACE from an attached Layer 2 ACL, detach the Layer 2 ACL on all the interfaces, modify the Layer 2 ACL, and re-attach it to the interfaces to recover the deleted ACE.
- Layer 2 ACL supports matching only on ISIS destination MAC address. It does not support any other Layer 2 fields, such as srcMAC, pcp etc. Configure any one of the ISIS Destination MAC addresses to drop ISIS packets. Non-ISIS destination MAC configuration is rejected.
- Hardware drops the ISIS packets when you configure one of the these destination MAC addresses:

- 01:80:c2:00:00:14
- 01:80:c2:00:00:15
- 09:00:2b:00:00:04
- 09:00:2b:00:00:05
- 01:00:5e:90:00:02
- 01:00:5e:90:00:03

You can configure only one set of L2 ACL configuration. Configure deny on ISSI DMAC first and then configure permit any any. If you configure any additional ACE, the configuration is rejected.

Configuration Example

Only the following L2 ACL configuration is allowed.

```
Router# configure
Router(config)# ethernet-services access-list isis-drop-all-l2-pdus
Router(config)# 5 remark Drain ISIS between two routers
Router(config)# 20 deny any host 0180.c200.0015
Router(config)# 200 permit any any
```

Perform the interface configuration:



Note You can configure this feature in bundle interfaces and subinterface, and physical interfaces and subinterfaces.

Perform the following steps to configure the feature in bundle interface.

```
Router# configure
Router(config)# interface Bundle-Ether 100 l2transport
Router(config-if)# mtu 2000
Router(config-if)# ethernet-services access-group isis-drop-all-l2-pdus ingress
```

Perform the following steps to configure the feature in bundle subinterface.

```
Router# configure
Router(config)# interface Bundle-Ether101.101 l2transport
Router(config-if)# encapsulation dot1q 101
Router(config-if)# rewrite ingress tag pop 1 symmetric
Router(config-if)# mtu 2000
Router(config-if)# ethernet-services access-group isis-drop-all-l2-pdus ingress
```

Perform the following steps to configure the feature in physical interface.

```
Router# configure
Router(config)# interface hundredGigE 0/0/0/0 l2transport
Router(config-if)# mtu 2000
Router(config-if)# ethernet-services access-group isis-drop-all-l2-pdus ingress
```

Perform the following steps to configure the feature in physical subinterface.

```
Router# configure
Router(config)# interface hundredGigE 0/3/0/1.100 l2transport
Router(config-if)# encapsulation dot1q 100
Router(config-if)# rewrite ingress tag pop 1 symmetric
```

```
Router(config-if)# mtu 2000
Router(config-if)# ethernet-services access-group isis-drop-all-l2-pdus ingress
```

Running Configuration

```
show running-config ethernet-services access-list
ethernet-services access-list isis-drop-all-l2-pdus
5 remark Drain ISIS between two routers
20 deny any host 0180.c200.0015
200 permit any any
```

```
Router# show running-config inter bundle-Ether 100
Sun Feb 14 12:51:27.425 PST
interface Bundle-Ether100 l2transport
mtu 2000
ethernet-services access-group isis-drop-all-l2-pdus ingress
!
```

```
Router# show running-config inter bundle-Ether 101.101
Sun Feb 14 12:51:27.425 PST
interface Bundle-Ether101.101 l2transport
encapsulation dot1q 101
rewrite ingress tag pop 1 symmetric
mtu 2000
ethernet-services access-group isis-drop-all-l2-pdus ingress
```

```
show running-config interface hundredGigE 0/0/0/0
Sun Feb 14 12:51:27.425 PST
interface hundredGigE 0/0/0/0 l2transport
mtu 2000
ethernet-services access-group isis-drop-all-l2-pdus ingress
!
```

```
show running-config interface hundredGigE 0/3/0/1.100
Sun Feb 14 12:51:27.425 PST
interface hundredGigE 0/3/0/1.100 l2transport
encapsulation dot1q 100
rewrite ingress tag pop 1 symmetric
mtu 2000
ethernet-services access-group isis-drop-all-l2-pdus ingress
!
```

Verification

```
Router# show access-lists ethernet-services l2 hardware ingress
location
Thu Jan 21 04:22:12.667 UTC
ethernet-services access-list l2
20 deny any host 0180.c200.0014 (1243345)
200 permit any any
```

```
Router# show access-lists ethernet-services
Sun Feb 14 12:52:09.539 PST
ethernet-services access-list isis-drop-all-l2-pdus
5 remark Drain ISIS between two routers.
20 deny any host 0180.c200.0015
200 permit any any
```

```
Router# show access-lists ethernet-services isis-drop-all-l2-pdus
hardware ingress location 0/0/CPU0
```

```
Sun Feb 14 12:52:39.620 PST
ethernet-services access-list isis-drop-all-l2-pdus
20 deny any host 0180.c200.0015
200 permit any any
```

```
Router# show access-lists ethernet-services isis-drop-all-l2-pdus
hardware ingress detail location 0/0/CPU0
Sun Feb 14 12:52:47.962 PST
```

```
isis-drop-all-l2-pdus Details:
Sequence Number: 20
NPU ID: 1
Number of DPA Entries: 1
ACL ID: 1
ACE Action: DENY
ACE Logging: DISABLED
Set TTL value: 0
Hit Packet Count: 0
Source MAC: 0000:0000:0000
Source MAC Mask: 0000:0000:0000
Destination MAC: 0180:C200:0015
Destination MAC Mask: FFFF:FFFF:FFFF
DPA Entry: 1
    Entry Index: 0
    DPA Handle: 0x93C84100
Sequence Number: 200
NPU ID: 1
Number of DPA Entries: 1
ACL ID: 1
ACE Action: PERMIT
ACE Logging: DISABLED
Set TTL value: 0
Source MAC: 0000:0000:0000
Source MAC Mask: 0000:0000:0000
Destination MAC: 0000:0000:0000
Destination MAC Mask: 0000:0000:0000
DPA Entry: 1
    Entry Index: 0
    DPA Handle: 0x93C84278
```

```
Router# show access-lists ethernet-services isis-drop-all-l2-pdus
hardware ingress sequence 20 location 0/0/CPU0
Sun Feb 14 12:53:46.456 PST
ethernet-services access-list isis-drop-all-l2-pdus
20 deny any host 0180.c200.0015
```

```
Router# show access-lists ethernet-services isis-drop-all-l2-pdus
hardware ingress sequence 20 detail location 0/0/CPU0
Sun Feb 14 12:54:14.849 PST
```

```
isis-drop-all-l2-pdus Details:
Sequence Number: 20
NPU ID: 1
Number of DPA Entries: 1
ACL ID: 1
ACE Action: DENY
ACE Logging: DISABLED
Set TTL value: 0
Hit Packet Count: 0
Source MAC: 0000:0000:0000
Source MAC Mask: 0000:0000:0000
Destination MAC: 0180:C200:0015
```

```
Destination MAC Mask: FFFF:FFFF:FFFF
DPA Entry: 1
  Entry Index: 0
  DPA Handle: 0x93C84100
```




CHAPTER 2

Implementing OSPF

Open Shortest Path First (OSPF) is an Interior Gateway Protocol (IGP) developed by the OSPF working group of the Internet Engineering Task Force (IETF). Designed expressly for IP networks, OSPF supports IP subnetting and tagging of externally derived routing information. OSPF also allows packet authentication when sending and receiving packets.

OSPF Version 3 (OSPFv3) expands on OSPF Version 2, providing support for IPv6 routing prefixes.

This module describes the concepts and tasks you need to implement both versions of OSPF on your software. The term “OSPF” implies both versions of the routing protocol, unless otherwise noted.

- [Prerequisites for Implementing OSPF](#) , on page 79
- [Information About Implementing OSPF](#) , on page 80
- [How to Implement OSPF](#) , on page 116
- [Configuring IP Fast Reroute Loop-free Alternate](#), on page 165
- [Configuration Examples for Implementing OSPF](#) , on page 169
- [Enable OSPF](#), on page 177
- [Verify OSPF Configuration and Operation](#), on page 180
- [Control Frequency That Same LSA Is Originated or Accepted for OSPF](#), on page 182
- [Create Virtual Link](#), on page 184
- [Summarize Subnetwork LSAs on OSPF ABR](#), on page 189
- [Graceful Shutdown for OSPFv3](#), on page 191
- [Enable Nonstop Routing for OSPFv2](#), on page 192
- [Warm Standby and Nonstop Routing for OSPF Version 3](#), on page 193
- [Enable Nonstop Routing for OSPFv3](#), on page 193
- [OSPFv2OSPF SPF Prefix Prioritization](#), on page 194
- [OSPFv3 Timers Update](#), on page 196
- [OSPF Link-State Database Overload Protection](#), on page 196

Prerequisites for Implementing OSPF

The following are prerequisites for implementing OSPF on Cisco IOS XR software:

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

- Configuration tasks for OSPFv3 assume that you are familiar with IPv6 addressing and basic configuration. See the *Implementing Network Stack IPv4 and IPv6 on* module of the for information on IPv6 routing and addressing.
- Before you enable OSPFv3 on an interface, you must perform the following tasks:
 - Complete the OSPF network strategy and planning for your IPv6 network. For example, you must decide whether multiple areas are required.
 - Enable IPv6 on the interface.
- Configuring authentication (IP Security) is an optional task. If you choose to configure authentication, you must first decide whether to configure plain text or Message Digest 5 (MD5) authentication, and whether the authentication applies to an entire area or specific interfaces.

Information About Implementing OSPF

To implement OSPF you need to understand the following concepts:

OSPF Functional Overview

OSPF is a routing protocol for IP. It is a link-state protocol, as opposed to a distance-vector protocol. A link-state protocol makes its routing decisions based on the states of the links that connect source and destination machines. The state of the link is a description of that interface and its relationship to its neighboring networking devices. The interface information includes the IP address of the interface, network mask, type of network to which it is connected, routers connected to that network, and so on. This information is propagated in various types of link-state advertisements (LSAs).

A router stores the collection of received LSA data in a link-state database. This database includes LSA data for the links of the router. The contents of the database, when subjected to the Dijkstra algorithm, extract data to create an OSPF routing table. The difference between the database and the routing table is that the database contains a complete collection of raw data; the routing table contains a list of shortest paths to known destinations through specific router interface ports.

OSPF is the IGP of choice because it scales to large networks. It uses areas to partition the network into more manageable sizes and to introduce hierarchy in the network. A router is attached to one or more areas in a network. All of the networking devices in an area maintain the same complete database information about the link states in their area only. They do not know about all link states in the network. The agreement of the database information among the routers in the area is called convergence.

At the intradomain level, OSPF can import routes learned using Intermediate System-to-Intermediate System (IS-IS). OSPF routes can also be exported into IS-IS. At the interdomain level, OSPF can import routes learned using Border Gateway Protocol (BGP). OSPF routes can be exported into BGP.

Unlike Routing Information Protocol (RIP), OSPF does not provide periodic routing updates. On becoming neighbors, OSPF routers establish an adjacency by exchanging and synchronizing their databases. After that, only changed routing information is propagated. Every router in an area advertises the costs and states of its links, sending this information in an LSA. This state information is sent to all OSPF neighbors one hop away. All the OSPF neighbors, in turn, send the state information unchanged. This flooding process continues until all devices in the area have the same link-state database.

To determine the best route to a destination, the software sums all of the costs of the links in a route to a destination. After each router has received routing information from the other networking devices, it runs the shortest path first (SPF) algorithm to calculate the best path to each destination network in the database.

The networking devices running OSPF detect topological changes in the network, flood link-state updates to neighbors, and quickly converge on a new view of the topology. Each OSPF router in the network soon has the same topological view again. OSPF allows multiple equal-cost paths to the same destination. Since all link-state information is flooded and used in the SPF calculation, multiple equal cost paths can be computed and used for routing.

On broadcast and nonbroadcast multiaccess (NBMA) networks, the designated router (DR) or backup DR performs the LSA flooding. On point-to-point networks, flooding simply exits an interface directly to a neighbor.

OSPF runs directly on top of IP; it does not use TCP or User Datagram Protocol (UDP). OSPF performs its own error correction by means of checksums in its packet header and LSAs.

In OSPFv3, the fundamental concepts are the same as OSPF Version 2, except that support is added for the increased address size of IPv6. New LSA types are created to carry IPv6 addresses and prefixes, and the protocol runs on an individual link basis rather than on an individual IP-subnet basis.

OSPF typically requires coordination among many internal routers: Area Border Routers (ABRs), which are routers attached to multiple areas, and Autonomous System Border Routers (ASBRs) that export reroutes from other sources (for example, IS-IS, BGP, or static routes) into the OSPF topology. At a minimum, OSPF-based routers or access servers can be configured with all default parameter values, no authentication, and interfaces assigned to areas. If you intend to customize your environment, you must ensure coordinated configurations of all routers.

Key Features Supported in the Cisco IOS XR Software OSPF Implementation

The Cisco IOS XR Software implementation of OSPF conforms to the OSPF Version 2 and OSPF Version 3 specifications detailed in the Internet RFC 2328 and RFC 2740, respectively.

The following key features are supported in the Cisco IOS XR Software implementation:

- Hierarchy—CLI hierarchy is supported.
- Inheritance—CLI inheritance is supported.
- Stub areas—Definition of stub areas is supported.
- NSF—Nonstop forwarding is supported.
- SPF throttling—Shortest path first throttling feature is supported.
- LSA throttling—LSA throttling feature is supported.
- Fast convergence—SPF and LSA throttle timers are set, configuring fast convergence. The OSPF LSA throttling feature provides a dynamic mechanism to slow down LSA updates in OSPF during network instability. LSA throttling also allows faster OSPF convergence by providing LSA rate limiting in milliseconds.
- Route redistribution—Routes learned using any IP routing protocol can be redistributed into any other IP routing protocol.
- Authentication—Plain text and MD5 authentication among neighboring routers within an area is supported.

- Routing interface parameters—Configurable parameters supported include interface output cost, retransmission interval, interface transmit delay, router priority, router “dead” and hello intervals, and authentication key.
- Virtual links—Virtual links are supported.
- Not-so-stubby area (NSSA)—RFC 1587 is supported.
- OSPF over demand circuit—RFC 1793 is supported.

Comparison of Cisco IOS XR Software OSPFv3 and OSPFv2

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

The key differences between the Cisco IOS XR Software OSPFv3 and OSPFv2 protocols are as follows:

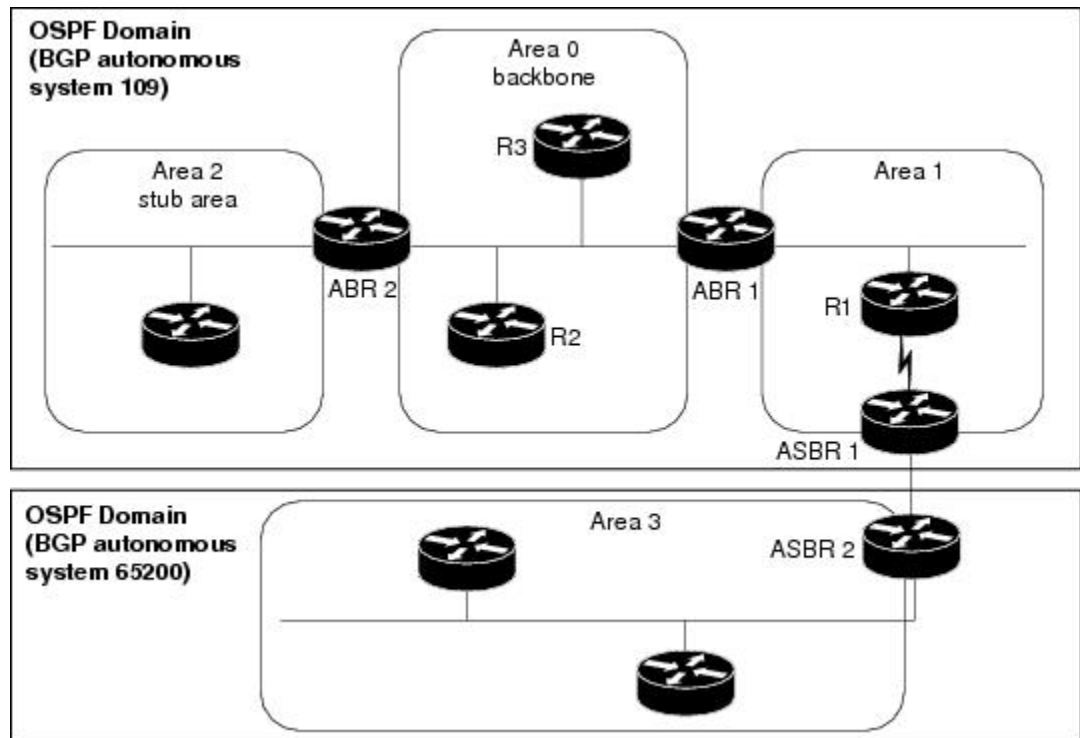
- OSPFv3 expands on OSPFv2 to provide support for IPv6 routing prefixes and the larger size of IPv6 addresses.
- When using an NBMA interface in OSPFv3, users must manually configure the router with the list of neighbors. Neighboring routers are identified by the link local address of the attached interface of the neighbor.
- Unlike in OSPFv2, multiple OSPFv3 processes can be run on a link.
- LSAs in OSPFv3 are expressed as “prefix and prefix length” instead of “address and mask.”
- The router ID is a 32-bit number with no relationship to an IPv6 address.

OSPF Routing Components

Before implementing OSPF, you must know what the routing components are and what purpose they serve. They consist of the autonomous system, area types, interior routers, ABRs, and ASBRs.

Figure 2: OSPF Routing Components

This figure illustrates the routing components in an OSPF network topology.



Autonomous Systems

The autonomous system is a collection of networks, under the same administrative control, that share routing information with each other. An autonomous system is also referred to as a routing domain. [#unique_93 unique_93_Connect_42_fig_1051926](#) shows two autonomous systems: 109 and 65200. An autonomous system can consist of one or more OSPF areas.

Areas

Areas allow the subdivision of an autonomous system into smaller, more manageable networks or sets of adjacent networks. As shown in [#unique_93 unique_93_Connect_42_fig_1051926](#), autonomous system 109 consists of three areas: Area 0, Area 1, and Area 2.

OSPF hides the topology of an area from the rest of the autonomous system. The network topology for an area is visible only to routers inside that area. When OSPF routing is within an area, it is called *intra-area routing*. This routing limits the amount of link-state information flood into the network, reducing routing traffic. It also reduces the size of the topology information in each router, conserving processing and memory requirements in each router.

Also, the routers within an area cannot see the detailed network topology outside the area. Because of this restricted view of topological information, you can control traffic flow between areas and reduce routing traffic when the entire autonomous system is a single routing domain.

Backbone Area

A backbone area is responsible for distributing routing information between multiple areas of an autonomous system. OSPF routing occurring outside of an area is called *interarea routing*.

The backbone itself has all properties of an area. It consists of ABRs, routers, and networks only on the backbone. As shown in [#unique_93 unique_93_Connect_42_fig_1051926](#), Area 0 is an OSPF backbone area. Any OSPF backbone area has a reserved area ID of 0.0.0.0.

Stub Area

A stub area is an area that does not accept route advertisements or detailed network information external to the area. A stub area typically has only one router that interfaces the area to the rest of the autonomous system. The stub ABR advertises a single default route to external destinations into the stub area. Routers within a stub area use this route for destinations outside the area and the autonomous system. This relationship conserves LSA database space that would otherwise be used to store external LSAs flooded into the area. In [#unique_93 unique_93_Connect_42_fig_1051926](#), Area 2 is a stub area that is reached only through ABR 2. Area 0 cannot be a stub area.

Not-so-Stubby Area

A Not-so-Stubby Area (NSSA) is similar to the stub area. NSSA does not flood Type 5 external LSAs from the core into the area, but can import autonomous system external routes in a limited fashion within the area.

NSSA allows importing of Type 7 autonomous system external routes within an NSSA area by redistribution. These Type 7 LSAs are translated into Type 5 LSAs by NSSA ABRs, which are flooded throughout the whole routing domain. Summarization and filtering are supported during the translation.

Use NSSA to simplify administration if you are a network administrator that must connect a central site using OSPF to a remote site that is using a different routing protocol.

Before NSSA, the connection between the corporate site border router and remote router could not be run as an OSPF stub area because routes for the remote site could not be redistributed into a stub area, and two routing protocols needed to be maintained. A simple protocol like RIP was usually run and handled the redistribution. With NSSA, you can extend OSPF to cover the remote connection by defining the area between the corporate router and remote router as an NSSA. Area 0 cannot be an NSSA.

Routers

The OSPF network is composed of ABRs, ASBRs, and interior routers.

Area Border Routers

An area border routers (ABR) is a router with multiple interfaces that connect directly to networks in two or more areas. An ABR runs a separate copy of the OSPF algorithm and maintains separate routing data for each area that is attached to, including the backbone area. ABRs also send configuration summaries for their attached areas to the backbone area, which then distributes this information to other OSPF areas in the autonomous system. In [#unique_93 unique_93_Connect_42_fig_1051926](#), there are two ABRs. ABR 1 interfaces Area 1 to the backbone area. ABR 2 interfaces the backbone Area 0 to Area 2, a stub area.

Autonomous System Boundary Routers (ASBR)

An autonomous system boundary router (ASBR) provides connectivity from one autonomous system to another system. ASBRs exchange their autonomous system routing information with boundary routers in other autonomous systems. Every router inside an autonomous system knows how to reach the boundary routers for its autonomous system.

ASBRs can import external routing information from other protocols like BGP and redistribute them as AS-external (ASE) Type 5 LSAs to the OSPF network. If the Cisco IOS XR router is an ASBR, you can

configure it to advertise VIP addresses for content as autonomous system external routes. In this way, ASBRs flood information about external networks to routers within the OSPF network.

ASBR routes can be advertised as a Type 1 or Type 2 ASE. The difference between Type 1 and Type 2 is how the cost is calculated. For a Type 2 ASE, only the external cost (metric) is considered when multiple paths to the same destination are compared. For a Type 1 ASE, the combination of the external cost and cost to reach the ASBR is used. Type 2 external cost is the default and is always more costly than an OSPF route and used only if no OSPF route exists.

Interior Routers

An interior router (such as R1 in [#unique_93 unique_93_Connect_42_fig_1051926](#)) is attached to one area (for example, all the interfaces reside in the same area).

OSPF Process and Router ID

An OSPF process is a logical routing entity running OSPF in a physical router. This logical routing entity should not be confused with the logical routing feature that allows a system administrator (known as the Cisco IOS XR Software Owner) to partition the physical box into separate routers.

A physical router can run multiple OSPF processes, although the only reason to do so would be to connect two or more OSPF domains. Each process has its own link-state database. The routes in the routing table are calculated from the link-state database. One OSPF process does not share routes with another OSPF process unless the routes are redistributed.

Each OSPF process is identified by a router ID. The router ID must be unique across the entire routing domain. OSPF obtains a router ID from the following sources, in order of decreasing preference:

- By default, when the OSPF process initializes, it checks if there is a router-id in the checkpointing database.
- The 32-bit numeric value specified by the OSPF router-id command in router configuration mode. (This value can be any 32-bit value. It is not restricted to the IPv4 addresses assigned to interfaces on this router, and need not be a routable IPv4 address.)
- The ITAL selected router-id.
- The primary IPv4 address of an interface over which this OSPF process is running. The first interface address in the OSPF interface is selected.

We recommend that the router ID be set by the **router-id** command in router configuration mode. Separate OSPF processes could share the same router ID, in which case they cannot reside in the same OSPF routing domain.

Supported OSPF Network Types

OSPF classifies different media into the following types of networks:

- NBMA networks
- Point-to-point networks (POS)
- Broadcast networks (Gigabit Ethernet)
- Point-to-multipoint

You can configure your Cisco IOS XR network as either a broadcast or an NBMA network. Using this feature, you can configure broadcast networks as NBMA networks when, for example, you have routers in your network that do not support multicast addressing.

Route Authentication Methods for OSPF

OSPF Version 2 supports two types of authentication: plain text authentication and MD5 authentication. By default, no authentication is enabled (referred to as null authentication in RFC 2178).

OSPF Version 3 supports all types of authentication except key rollover.

Plain Text Authentication

Plain text authentication (also known as Type 1 authentication) uses a password that travels on the physical medium and is easily visible to someone that does not have access permission and could use the password to infiltrate a network. Therefore, plain text authentication does not provide security. It might protect against a faulty implementation of OSPF or a misconfigured OSPF interface trying to send erroneous OSPF packets.

MD5 Authentication

MD5 authentication provides a means of security. No password travels on the physical medium. Instead, the router uses MD5 to produce a message digest of the OSPF packet plus the key, which is sent on the physical medium. Using MD5 authentication prevents a router from accepting unauthorized or deliberately malicious routing updates, which could compromise your network security by diverting your traffic.



Note MD5 authentication supports multiple keys, requiring that a key number be associated with a key.

See [#unique_107](#).

Authentication Strategies

Authentication can be specified for an entire process or area, or on an interface or a virtual link. An interface or virtual link can be configured for only one type of authentication, not both. Authentication configured for an interface or virtual link overrides authentication configured for the area or process.

If you intend for all interfaces in an area to use the same type of authentication, you can configure fewer commands if you use the **authentication** command in the area configuration submode (and specify the **message-digest** keyword if you want the entire area to use MD5 authentication and HMAC SHA 256 authentication). This strategy requires fewer commands than specifying authentication for each interface.

Key Rollover

To support the changing of an MD5 key in an operational network without disrupting OSPF adjacencies (and hence the topology), a key rollover mechanism is supported. As a network administrator configures the new key into the multiple networking devices that communicate, some time exists when different devices are using both a new key and an old key. If an interface is configured with a new key, the software sends two copies of the same packet, each authenticated by the old key and new key. The software tracks which devices start using the new key, and the software stops sending duplicate packets after it detects that all of its neighbors are using the new key. The software then discards the old key. The network administrator must then remove the old key from each the configuration file of each router.

Neighbors and Adjacency for OSPF

Routers that share a segment (Layer 2 link between two interfaces) become neighbors on that segment. OSPF uses the hello protocol as a neighbor discovery and keep alive mechanism. The hello protocol involves receiving and periodically sending hello packets out each interface. The hello packets list all known OSPF neighbors on the interface. Routers become neighbors when they see themselves listed in the hello packet of the neighbor. After two routers are neighbors, they may proceed to exchange and synchronize their databases, which creates an adjacency. On broadcast and NBMA networks all neighboring routers have an adjacency.

OSPF strict-mode Support for BFD Dampening

Strict-mode is an OSPF BFD operation mode which keeps the neighbor in a down state until the BFD session is up. The status of the neighbor node shows as awaiting BFD session up in the output of the **show ospf neighbor** command. This will ensure that client protocols do not operate independent of the declared state of BFD.

Restrictions

- Strict-mode and non-strict-mode modes of operation are incompatible and will cause OSPF to never form a neighbor relationship. Strict-mode can not be configured on one node and default/non-strict mode on the other. Both BFD neighbors must run IOS-XR images that support strict-mode. However, if by design the additional BFD clients have already initiated the BFD session and OSPF is not the only initiator, then they may form a neighbor relationship.
- Due to the dependency on BFD, OSPF operating in strict-mode may experience delayed neighbor establishment and full adjacency.

Enabling strict-mode

Strict-mode can be enabled at different levels - process, area and interface. The following procedure describes how to enable BFD strict-mode for Open Shortest Path First (OSPF) on an interface:

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*
4. **interface** *type interface-path-id*
5. **bfd fast-detect strict-mode**
6. **commit**
7. **show ospf interface** *type interface-path-id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters global configuration mode.

	Command or Action	Purpose
Step 2	router ospf <i>process-name</i> Example: RP/0/(config)# router ospf 1	Enters OSPF configuration mode, allowing you to configure the OSPF routing process. Use the show ospf command in <code>mode</code> to obtain the process-name for the current router.
Step 3	area <i>area-id</i> Example: RP/0/(config-ospf)# area 0	Configures an Open Shortest Path First (OSPF) area. Replace <i>area-id</i> with the OSPF area identifier.
Step 4	interface <i>type interface-path-id</i> Example: RP/0/(config-ospf-ar)# interface gigabitEthernet 0/3/0/1	Enters interface configuration mode and specifies the interface name and notation <i>rack/slot/module/port</i> . The example indicates a Gigabit Ethernet interface in modular services card slot 3.
Step 5	bfd fast-detect strict-mode Example: RP/0/(config-ospf-ar-if)# bfd fast-detect strict-mode	Enables strict-mode to hold down neighbor session until BFD session is up.
Step 6	commit	Commits the changes to the running configuration.
Step 7	show ospf interface <i>type interface-path-id</i> Example: RP/0/(config-ospf-ar-if)# show ospf interface gigabitEthernet 0/3/0/1	Verify that strict-mode is enabled on the appropriate interface.

BFD strict-mode: Example

The following example shows how to enable BFD strict-mode for OSPF on a Gigabit Ethernet interface and check the OSPF interface information. The value of **Mode** displays as **Strict** when BFD strict-mode is enabled. By default, the value of **Mode** displays as **Default**.

```
RP/0/#configure
RP/0/(config)#router ospf 0
RP/0/(config-ospf)#area 0
RP/0/(config-ospf-ar)#interface gigabitEthernet 0/3/0/1
RP/0/(config-ospf-ar-if)#bfd fast-detect strict-mode
RP/0/(config-ospf-ar-if)#commit
RP/0/(config-ospf-ar-if)#end
RP/0/#show ospf interface gigabitEthernet 0/3/0/1
```

```
GigabitEthernet0/3/0/1 is up, line protocol is up
Internet Address 10.1.1.2/24, Area 0
Process ID 1, Router ID 2.2.2.2, Network Type BROADCAST, Cost: 1
Transmit Delay is 1 sec, State DR, Priority 1, MTU 1500, MaxPktSz 1500
BFD enabled, BFD interval 150 msec, BFD multiplier 3, Mode: Strict
Designated Router (ID) 2.2.2.2, Interface address 10.1.1.2
No backup designated router on this network
```



```

Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
  Hello due in 00:00:07:358
Index 1/1, flood queue length 0
Next 0(0)/0(0)
Last flood scan length is 1, maximum is 1
Last flood scan time is 0 msec, maximum is 0 msec
LS Ack List: current length 0, high water mark 1
Neighbor Count is 1, Adjacent neighbor count is 0
Suppress hello for 0 neighbor(s)
Multi-area interface Count is 0

```

The following example shows the output of the **show ospf neighbor** command. # indicates that the neighbor is waiting for the BFD session to come up.

```
RP/0/#show ospf neighbor
```

```
Neighbors for OSPF 1
```

Neighbor ID	Pri	State	Dead Time	Address	Interface
1.1.1.1	0	DOWN/DROTHER	00:00:33	10.1.1.3/24	GigabitEthernet0/3/0/1#

```
Total neighbor count: 1
```

OSPF FIB Download Notification

OSPF FIB Download Notification feature minimizes the ingress traffic drop for a prolonged period of time after the line card reloads and this feature is enabled by default.

Open Shortest Path First (OSPF) registers with Routing Information Base (RIB) through Interface Table Attribute Library (ITAL) which keeps the interface down until all the routes are downloaded to Forwarding Information Base (FIB). OSPF gets the Interface Up notification when all the routes on the reloaded line card are downloaded through RIB/FIB.

RIB provides notification to registered clients when a:

- Node is lost.
- Node is created.
- Node's FIB upload is completed.

Designated Router (DR) for OSPF

On point-to-point and point-to-multipoint networks, the Cisco IOS XR software floods routing updates to immediate neighbors. No DR or backup DR (BDR) exists; all routing information is flooded to each router.

On broadcast or NBMA segments only, OSPF minimizes the amount of information being exchanged on a segment by choosing one router to be a DR and one router to be a BDR. Thus, the routers on the segment have a central point of contact for information exchange. Instead of each router exchanging routing updates with every other router on the segment, each router exchanges information with the DR and BDR. The DR and BDR relay the information to the other routers. On broadcast network segments the number of OSPF packets is further reduced by the DR and BDR sending such OSPF updates to a multicast IP address that all OSPF routers on the network segment are listening on.

The software looks at the priority of the routers on the segment to determine which routers are the DR and BDR. The router with the highest priority is elected the DR. If there is a tie, then the router with the higher

router ID takes precedence. After the DR is elected, the BDR is elected the same way. A router with a router priority set to zero is ineligible to become the DR or BDR.

Default Route for OSPF

Type 5 (ASE) LSAs are generated and flooded to all areas except stub areas. For the routers in a stub area to be able to route packets to destinations outside the stub area, a default route is injected by the ABR attached to the stub area.

The cost of the default route is 1 (default) or is determined by the value specified in the **default-cost** command.

Link-State Advertisement Types for OSPF Version 2

Each of the following LSA types has a different purpose:

- Router LSA (Type 1)—Describes the links that the router has within a single area, and the cost of each link. These LSAs are flooded within an area only. The LSA indicates if the router can compute paths based on quality of service (QoS), whether it is an ABR or ASBR, and if it is one end of a virtual link. Type 1 LSAs are also used to advertise stub networks.
- Network LSA (Type 2)—Describes the link state and cost information for all routers attached to a multiaccess network segment. This LSA lists all the routers that have interfaces attached to the network segment. It is the job of the designated router of a network segment to generate and track the contents of this LSA.
- Summary LSA for ABRs (Type 3)—Advertises internal networks to routers in other areas (interarea routes). Type 3 LSAs may represent a single network or a set of networks aggregated into one prefix. Only ABRs generate summary LSAs.
- Summary LSA for ASBRs (Type 4)—Advertises an ASBR and the cost to reach it. Routers that are trying to reach an external network use these advertisements to determine the best path to the next hop. ABRs generate Type 4 LSAs.
- Autonomous system external LSA (Type 5)—Redistributes routes from another autonomous system, usually from a different routing protocol into OSPF.
- Autonomous system external LSA (Type 7)—Provides for carrying external route information within an NSSA. Type 7 LSAs may be originated by and advertised throughout an NSSA. NSSAs do not receive or originate Type 5 LSAs. Type 7 LSAs are advertised only within a single NSSA. They are not flooded into the backbone area or into any other area by border routers.
- Intra-area-prefix LSAs (Type 9)—A router can originate multiple intra-area-prefix LSAs for every router or transit network, each with a unique link-state ID. The link-state ID for each intra-area-prefix LSA describes its association to either the router LSA or network LSA and contains prefixes for stub and transit networks.
- Area local scope (Type 10)—Opaque LSAs are not flooded past the borders of their associated area.
- Link-state (Type 11)—The LSA is flooded throughout the AS. The flooding scope of Type 11 LSAs are equivalent to the flooding scope of AS-external (Type 5) LSAs. Similar to Type 5 LSAs, the LSA is rejected if a Type 11 opaque LSA is received in a stub area from a neighboring router within the stub area. Type 11 opaque LSAs have these attributes:
 - LSAs are flooded throughout all transit areas.

- LSAs are not flooded into stub areas from the backbone.
- LSAs are not originated by routers into their connected stub areas.

Link-State Advertisement Types for OSPFv3

Each of the following LSA types has a different purpose:

- Router LSA (Type 1)—Describes the link state and costs of a the router link to the area. These LSAs are flooded within an area only. The LSA indicates whether the router is an ABR or ASBR and if it is one end of a virtual link. Type 1 LSAs are also used to advertise stub networks. In OSPFv3, these LSAs have no address information and are network protocol independent. In OSPFv3, router interface information may be spread across multiple router LSAs. Receivers must concatenate all router LSAs originated by a given router before running the SPF calculation.
- Network LSA (Type 2)—Describes the link state and cost information for all routers attached to a multiaccess network segment. This LSA lists all OSPF routers that have interfaces attached to the network segment. Only the elected designated router for the network segment can generate and track the network LSA for the segment. In OSPFv3, network LSAs have no address information and are network-protocol-independent.
- Interarea-prefix LSA for ABRs (Type 3)—Advertises internal networks to routers in other areas (interarea routes). Type 3 LSAs may represent a single network or set of networks aggregated into one prefix. Only ABRs generate Type 3 LSAs. In OSPFv3, addresses for these LSAs are expressed as “prefix and prefix length” instead of “address and mask.” The default route is expressed as a prefix with length 0.
- Interarea-router LSA for ASBRs (Type 4)—Advertises an ASBR and the cost to reach it. Routers that are trying to reach an external network use these advertisements to determine the best path to the next hop. ABRs generate Type 4 LSAs.
- Autonomous system external LSA (Type 5)—Redistributes routes from another autonomous system, usually from a different routing protocol into OSPF. In OSPFv3, addresses for these LSAs are expressed as “prefix and prefix length” instead of “address and mask.” The default route is expressed as a prefix with length 0.
- Autonomous system external LSA (Type 7)—Provides for carrying external route information within an NSSA. Type 7 LSAs may be originated by and advertised throughout an NSSA. NSSAs do not receive or originate Type 5 LSAs. Type 7 LSAs are advertised only within a single NSSA. They are not flooded into the backbone area or into any other area by border routers.
- Link LSA (Type 8)—Has link-local flooding scope and is never flooded beyond the link with which it is associated. Link LSAs provide the link-local address of the router to all other routers attached to the link or network segment, inform other routers attached to the link of a list of IPv6 prefixes to associate with the link, and allow the router to assert a collection of Options bits to associate with the network LSA that is originated for the link.
- Intra-area-prefix LSAs (Type 9)—A router can originate multiple intra-area-prefix LSAs for every router or transit network, each with a unique link-state ID. The link-state ID for each intra-area-prefix LSA describes its association to either the router LSA or network LSA and contains prefixes for stub and transit networks.

An address prefix occurs in almost all newly defined LSAs. The prefix is represented by three fields: Prefix Length, Prefix Options, and Address Prefix. In OSPFv3, addresses for these LSAs are expressed as “prefix and prefix length” instead of “address and mask.” The default route is expressed as a prefix with length 0.

Inter-area-prefix and intra-area-prefix LSAs carry all IPv6 prefix information that, in IPv4, is included in router LSAs and network LSAs. The Options field in certain LSAs (router LSAs, network LSAs, interarea-router LSAs, and link LSAs) has been expanded to 24 bits to provide support for OSPF in IPv6.

In OSPFv3, the sole function of link-state ID in interarea-prefix LSAs, interarea-router LSAs, and autonomous system external LSAs is to identify individual pieces of the link-state database. All addresses or router IDs that are expressed by the link-state ID in OSPF Version 2 are carried in the body of the LSA in OSPFv3.

Virtual Link and Transit Area for OSPF

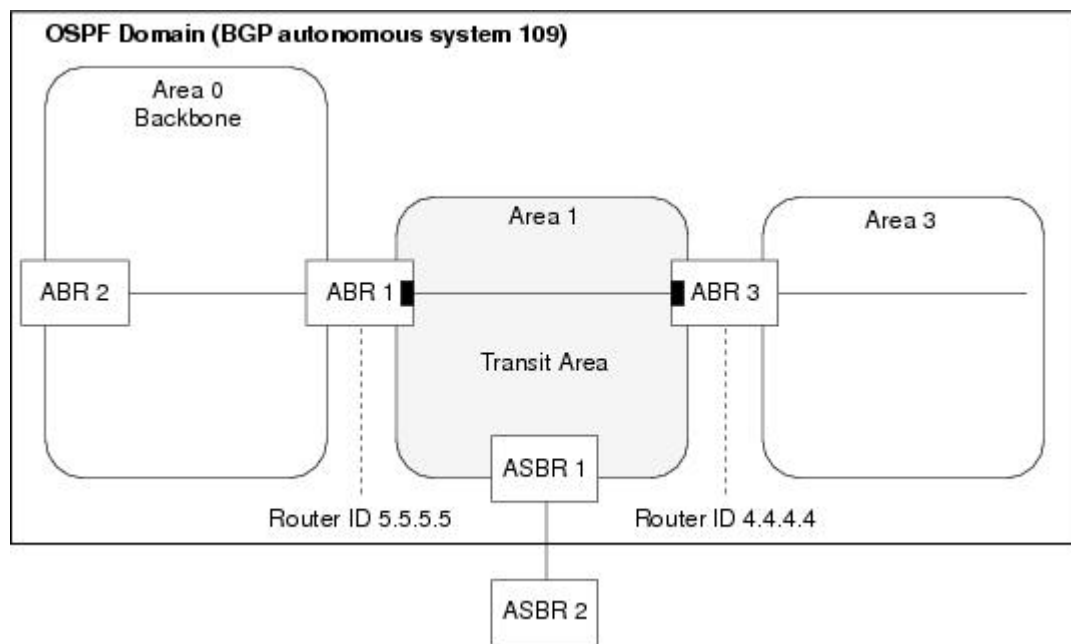
In OSPF, routing information from all areas is first summarized to the backbone area by ABRs. The same ABRs, in turn, propagate such received information to their attached areas. Such hierarchical distribution of routing information requires that all areas be connected to the backbone area (Area 0). Occasions might exist for which an area must be defined, but it cannot be physically connected to Area 0. Examples of such an occasion might be if your company makes a new acquisition that includes an OSPF area, or if Area 0 itself is partitioned.

In the case in which an area cannot be connected to Area 0, you must configure a virtual link between that area and Area 0. The two endpoints of a virtual link are ABRs, and the virtual link must be configured in both routers. The common nonbackbone area to which the two routers belong is called a transit area. A virtual link specifies the transit area and the router ID of the other virtual endpoint (the other ABR).

A virtual link cannot be configured through a stub area or NSSA.

Figure 3: Virtual Link to Area 0

This figure illustrates a virtual link from Area 3 to Area 0.



Passive Interface

Setting an interface as passive disables the sending of routing updates for the neighbors, hence adjacencies will not be formed in OSPF. However, the particular subnet will continue to be advertised to OSPF neighbors. Use the **passive** command in appropriate mode to suppress the sending of OSPF protocol operation on an interface.

It is recommended to use passive configuration on interfaces that are connecting LAN segments with hosts to the rest of the network, but are not meant to be transit links between routers.

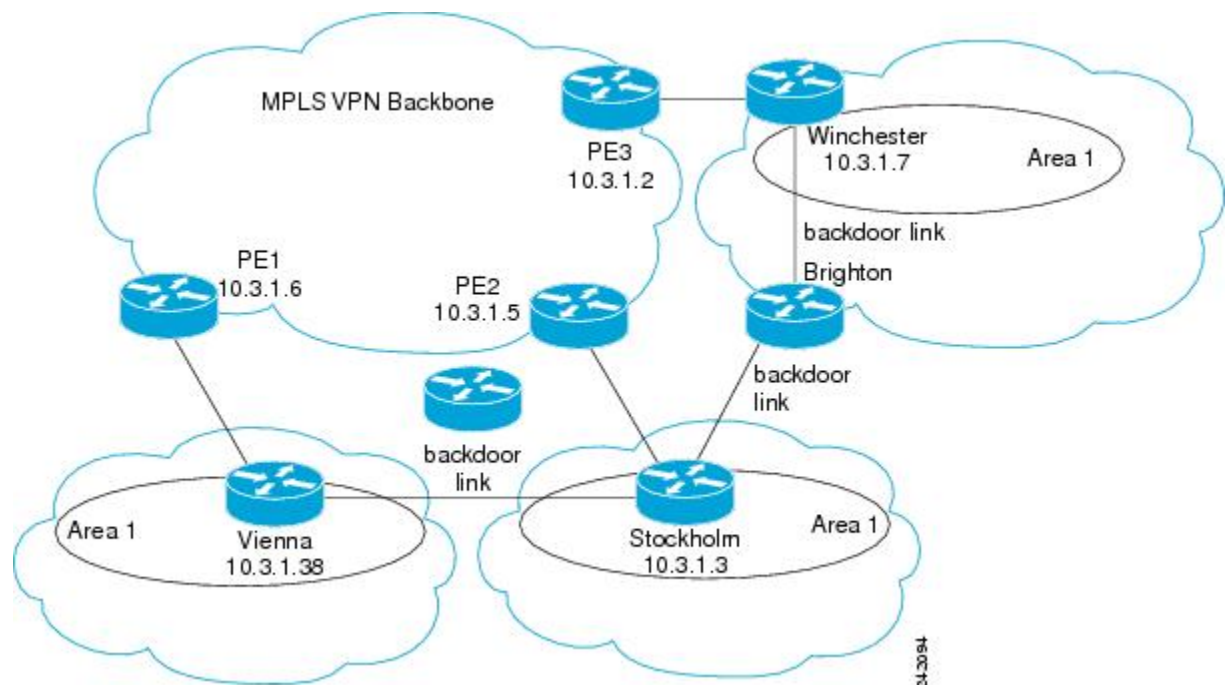
OSPFv2 Sham Link Support for MPLS VPN

In an MPLS VPN environment, several VPN client sites can be connected in the same OSPF area. If these sites are connected over a backdoor link (intra-area link) and connected over the VPN backbone, all traffic passes over the backdoor link instead of over the VPN backbone, because provider edge routers advertise OSPF routes learned over the VPN backbone as inter-area or external routes that are less preferred than intra-area routes advertised over backdoor links.

To correct this default OSPF behavior in an MPLS VPN, configure a sham link between two provider edge (PE) routers to connect the sites through the MPLS VPN backbone. A sham link represents an intra-area (unnumbered point-to-point) connection between PE routers. All other routers in the area see the sham link and use it to calculate intra-area shortest path first (SPF) routes to the remote site. A cost must be configured with each sham link to determine whether traffic is sent over the backdoor link or sham link.

Configured source and destination addresses serve as the endpoints of the sham link. The source and destination IP addresses must belong to the VRF and must be advertised by Border Gateway Protocol (BGP) as host routes to remote PE routers. The sham-link endpoint addresses should not be advertised by OSPF.

Figure 4: Backdoor Paths Between OSPF Client Sites

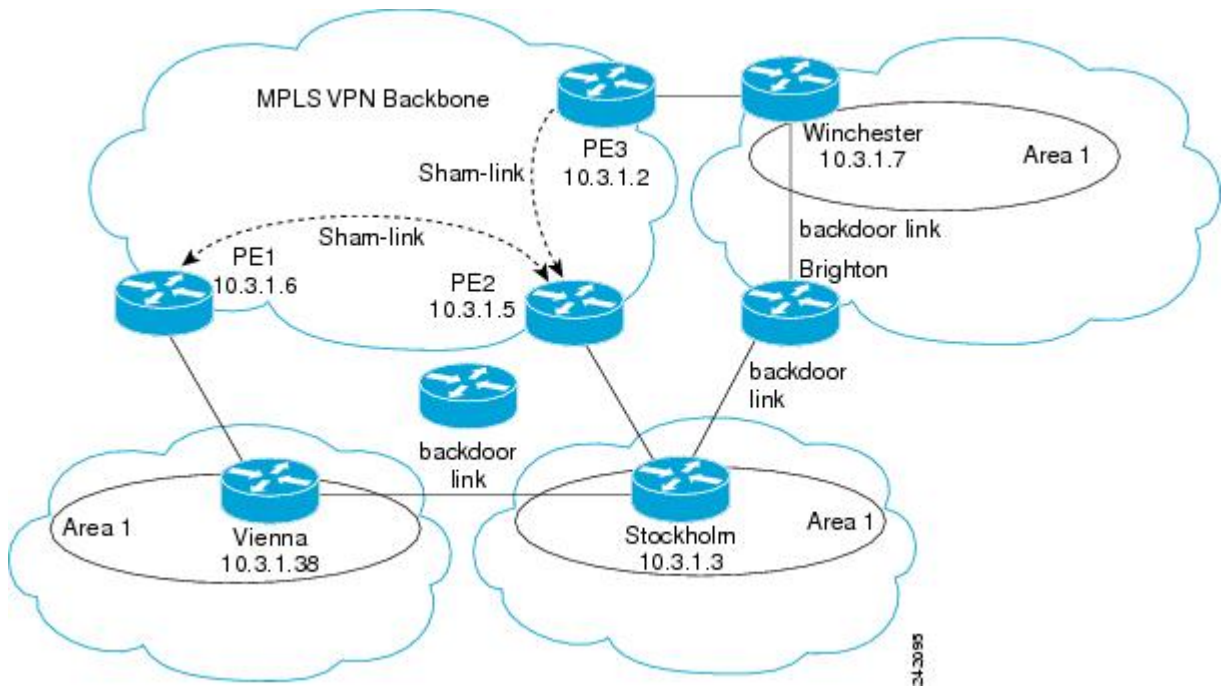


For example, the figure above shows three client sites, each with backdoor links. Because each site runs OSPF within Area 1 configuration, all routing between the sites follows the intra-area path across the backdoor links instead of over the MPLS VPN backbone.

If the backdoor links between the sites are used only for backup purposes, default route selection over the backbone link is not acceptable as it creates undesirable traffic flow. To establish the desired path selection over the MPLS backbone, an additional OSPF intra-area (sham link) link between the ingress and egress PE routers must be created.

A sham link is required between any two VPN sites that belong to the same OSPF area and share an OSPF backdoor link. If no backdoor link exists between sites, no sham link is required.

Figure 5: Sham Link Between PE Routers to Connected OSPF Client Sites



The figure above shows an MPLS VPN topology where a sham link configuration is necessary. A VPN client has three sites, each with a backdoor link. Two sham links are configured, one between PE-1 and PE-2 and another between PE-2 and PE-3. A sham link is not required between PE-1 and PE-3, because there is no backdoor link between these sites.

When a sham link is configured between the PE routers, the PE routers can populate the virtual routing and forwarding (VRF) table with the OSPF routes learned over the sham link. These OSPF routes have a larger administrative distance than BGP routes. If BGP routes are available, they are preferred over these OSPF routes with the high administrative distance.

OSPFv3 Sham Link Support for MPLS VPN

OSPFv3 sham link represents the VPN backbone as a single point-to-point connection between the two PEs. OSPFv3 treats the sham link as a point-to-point unnumbered interface, similar to virtual-link. When OSPFv3 sham link is configured, ensure that the route to the remote endpoint of the sham-link exists in the VRF RIB.

If the route to the remote endpoint exists, sham link interface is brought up. If the route to the remote endpoint of the sham-link is removed from the VRF RIB, OSPFv3 receives redistribution callback and brings the sham link down.

Graceful Restart Procedure over the Sham-link

OSPFv3 treats the sham link as any other interface during the switch-over or process restart. OSPFv3 assumes that all the configured sham links are UP and tries to form an adjacency over them.

If the sham link is down prior to the switch-over, OSPFv3 sends the Hello packets to the remote endpoint. Once the final convergence signal is received from the RIB, OSPFv3 keeps the sham link either up or down based on the BGP route for each configured sham link in the RIB.

OSPFv3 installs the high AD routes over the sham link only after the BGP convergence is complete.

ECMP and OSPFv3 Sham-link

Equal Cost Multipath (ECMP) mechanism is used to load-balance traffic on the Sham-link if there are multiple iBGP path for a prefix. If the sham link path and the backdoor path have the same cost, ECMP between the sham link path and backdoor path is not supported.

OSPFv2OSPF SPF Prefix Prioritization

The OSPFv2 OSPF SPF Prefix Prioritization feature enables an administrator to converge, in a faster mode, important prefixes during route installation.

When a large number of prefixes must be installed in the Routing Information Base (RIB) and the Forwarding Information Base (FIB), the update duration between the first and last prefix, during SPF, can be significant.

In networks where time-sensitive traffic (for example, VoIP) may transit to the same router along with other traffic flows, it is important to prioritize RIB and FIB updates during SPF for these time-sensitive prefixes.

The OSPFv2OSPF SPF Prefix Prioritization feature provides the administrator with the ability to prioritize important prefixes to be installed, into the RIB during SPF calculations. Important prefixes converge faster among prefixes of the same route type per area. Before RIB and FIB installation, routes and prefixes are assigned to various priority batch queues in the OSPF local RIB, based on specified route policy. The RIB priority batch queues are classified as "critical," "high," "medium," and "low," in the order of decreasing priority.

When enabled, prefix alters the sequence of updating the RIB with this prefix priority:

Critical > High > Medium > Low

As soon as prefix priority is configured, /32 prefixes are no longer preferred by default; they are placed in the low-priority queue, if they are not matched with higher-priority policies. Route policies must be devised to retain /32s in the higher-priority queues (high-priority or medium-priority queues).

Priority is specified using route policy, which can be matched based on IP addresses or route tags. During SPF, a prefix is checked against the specified route policy and is assigned to the appropriate RIB batch priority queue.

These are examples of this scenario:

- If only high-priority route policy is specified, and no route policy is configured for a medium priority:
 - Permitted prefixes are assigned to a high-priority queue.

- Unmatched prefixes, including /32s, are placed in a low-priority queue.
- If both high-priority and medium-priority route policies are specified, and no maps are specified for critical priority:
 - Permitted prefixes matching high-priority route policy are assigned to a high-priority queue.
 - Permitted prefixes matching medium-priority route policy are placed in a medium-priority queue.
 - Unmatched prefixes, including /32s, are moved to a low-priority queue.
- If both critical-priority and high-priority route policies are specified, and no maps are specified for medium priority:
 - Permitted prefixes matching critical-priority route policy are assigned to a critical-priority queue.
 - Permitted prefixes matching high-priority route policy are assigned to a high-priority queue.
 - Unmatched prefixes, including /32s, are placed in a low-priority queue.
- If only medium-priority route policy is specified and no maps are specified for high priority or critical priority:
 - Permitted prefixes matching medium-priority route policy are assigned to a medium-priority queue.
 - Unmatched prefixes, including /32s, are placed in a low-priority queue.

Use the **[no] spf prefix-priority route-policy** *rpl* command to prioritize OSPFv2OSPF prefix installation into the global RIB during SPF.

SPF prefix prioritization is disabled by default. In disabled mode, /32 prefixes are installed into the global RIB, before other prefixes. If SPF prioritization is enabled, routes are matched against the route-policy criteria and are assigned to the appropriate priority queue based on the SPF priority set. Unmatched prefixes, including /32s, are placed in the low-priority queue.

If all /32s are desired in the high-priority queue or medium-priority queue, configure this single route map:

```
prefix-set ospf-medium-prefixes
0.0.0.0/0 ge 32
end-set
```

Route Redistribution for OSPF

Redistribution allows different routing protocols to exchange routing information. This technique can be used to allow connectivity to span multiple routing protocols. It is important to remember that the **redistribute** command controls redistribution *into* an OSPF process and not from OSPF. See [Configuration Examples for Implementing OSPF](#), on page 169 for an example of route redistribution for OSPF.

OSPF Shortest Path First Throttling

OSPF SPF throttling makes it possible to configure SPF scheduling in millisecond intervals and to potentially delay SPF calculations during network instability. SPF is scheduled to calculate the Shortest Path Tree (SPT) when there is a change in topology. One SPF run may include multiple topology change events.

The interval at which the SPF calculations occur is chosen dynamically and based on the frequency of topology changes in the network. The chosen interval is within the boundary of the user-specified value ranges. If network topology is unstable, SPF throttling calculates SPF scheduling intervals to be longer until topology becomes stable.

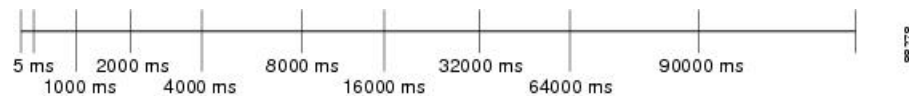
SPF calculations occur at the interval set by the **timers throttle spf** command. The wait interval indicates the amount of time to wait until the next SPF calculation occurs. Each wait interval after that calculation is twice as long as the previous interval until the interval reaches the maximum wait time specified.

The SPF timing can be better explained using an example. In this example, the start interval is set at 5 milliseconds (ms), initial wait interval at 1000 ms, and maximum wait time at 90,000 ms.

```
timers spf 5 1000 90000
```

Figure 6: SPF Calculation Intervals Set by the timers spf Command

This figure shows the intervals at which the SPF calculations occur as long as at least one topology change event is received in a given wait interval.

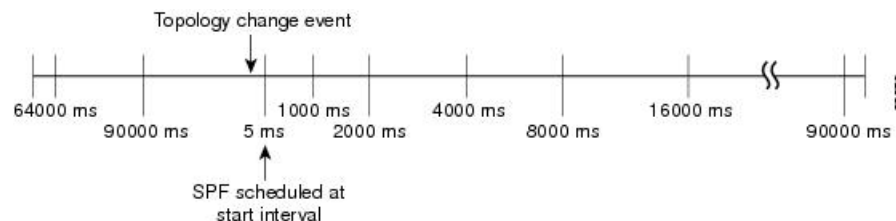


Notice that the wait interval between SPF calculations doubles when at least one topology change event is received during the previous wait interval. After the maximum wait time is reached, the wait interval remains the same until the topology stabilizes and no event is received in that interval.

If the first topology change event is received after the current wait interval, the SPF calculation is delayed by the amount of time specified as the start interval. The subsequent wait intervals continue to follow the dynamic pattern.

If the first topology change event occurs after the maximum wait interval begins, the SPF calculation is again scheduled at the start interval and subsequent wait intervals are reset according to the parameters specified in the **timers throttle spf** command. Notice in the following figure that a topology change event was received after the start of the maximum wait time interval and that the SPF intervals have been reset.

Figure 7: Timer Intervals Reset After Topology Change Event



Nonstop Forwarding for OSPF Version 2

Cisco IOS XR Software NSF for OSPF Version 2 allows for the forwarding of data packets to continue along known routes while the routing protocol information is being restored following a process restart or failover. With NSF, peer networking devices do not experience routing flaps. During process restart or failover, data traffic is forwarded through intelligent line cards while the standby Route Processor (RP) assumes control

from the failed RP. The ability of line cards to remain up through a process restart, and to be kept current with the Forwarding Information Base (FIB) on the active RP is key to Cisco IOS XR Software NSF operation.

Routing protocols, such as OSPF, run only on the active RP or DRP and receive routing updates from their neighbor routers. When an OSPF NSF-capable router performs a process restart, it must perform two tasks to resynchronize its link-state database with its OSPF neighbors. First, it must relearn the available OSPF neighbors on the network without causing a reset of the neighbor relationship. Second, it must reacquire the contents of the link-state database for the network.

As quickly as possible after an RP failover or process restart, the NSF-capable router sends an OSPF NSF signal to neighboring NSF-aware devices. This signal is in the form of a link-local LSA generated by the failed-over router. Neighbor networking devices recognize this signal as a cue that the neighbor relationship with this router should not be reset. As the NSF-capable router receives signals from other routers on the network, it can begin to rebuild its neighbor list.

After neighbor relationships are reestablished, the NSF-capable router begins to resynchronize its database with all of its NSF-aware neighbors. At this point, the routing information is exchanged between the OSPF neighbors. After this exchange is completed, the NSF-capable device uses the routing information to remove stale routes, update the RIB, and update the FIB with the new forwarding information. OSPF on the router and the OSPF neighbors are now fully converged.

Graceful Shutdown for OSPFv3

The OSPFv3 Graceful Shutdown feature preserves the data plane capability in these circumstances:

- RP failure resulting in a switch-over to the backup processor
- Planned OSPFv3 process restart, such as a restart resulting from a software upgrade or downgrade
- Unplanned OSPFv3 process restart, such as a restart resulting from a process crash

In addition, OSPFv3 will unilaterally shutdown and enter the exited state when a critical memory event, indicating the processor is critically low on available memory, is received from the sysmon watch dog process.

This feature supports nonstop data forwarding on established routes while the OSPFv3 routing protocol restarts. Therefore, this feature enhances high availability of IPv6 forwarding.

Modes of Graceful Restart Operation

The operational modes that a router can be in for this feature are restart mode and helper mode. Restart mode occurs when the OSPFv3 process is doing a graceful restart. Helper mode refers to the neighbor routers that continue to forward traffic on established OSPFv3 routes while OSPFv3 is restarting on a neighboring router.

Restart Mode

When the OSPFv3 process starts up, it determines whether it must attempt a graceful restart. The determination is based on whether graceful restart was previously enabled. (OSPFv3 does not attempt a graceful restart upon the first-time startup of the router.) When OSPFv3 graceful restart is enabled, it changes the purge timer in the RIB to a nonzero value. See [Configuring OSPFv3 Graceful Restart, on page 141](#), for descriptions of how to enable and configure graceful restart.

During a graceful restart, the router does not populate OSPFv3 routes in the RIB. It tries to bring up full adjacencies with the fully adjacent neighbors that OSPFv3 had before the restart. Eventually, the OSPFv3 process indicates to the RIB that it has converged, either for the purpose of terminating the graceful restart (for any reason) or because it has completed the graceful restart.

The following are general details about restart mode. More detailed information on behavior and certain restrictions and requirements appears in [Graceful Restart Requirements and Restrictions, on page 100](#) section.

- If OSPFv3 attempts a restart too soon after the most recent restart, the OSPFv3 process is most likely crashing repeatedly, so the new graceful restart stops running. To control the period between allowable graceful restarts, use the **graceful-restart interval** command.
- When OSPFv3 starts a graceful restart with the first interface that comes up, a timer starts running to limit the duration (or lifetime) of the graceful restart. You can configure this period with the **graceful-restart lifetime** command. On each interface that comes up, a *grace* LSA (Type 11) is flooded to indicate to the neighboring routers that this router is attempting graceful restart. The neighbors enter into helper mode.
- The designated router and backup designated router check of the hello packet received from the restarting neighbor is bypassed, because it might not be valid.

Helper Mode

Helper mode is enabled by default. When a (helper) router receives a grace LSA (Type 11) from a router that is attempting a graceful restart, the following events occur:

- If helper mode has been disabled through the **graceful-restart helper disable** command, the router drops the LSA packet.
- If helper mode is enabled, the router enters helper mode if all of the following conditions are met:
 - The local router itself is not attempting a graceful restart.
 - The local (helping) router has full adjacency with the sending neighbor.
 - The value of *lsage* (link state age) in the received LSA is less than the requested grace period.
 - The sender of the grace LSA is the same as the originator of the grace LSA.
- Upon entering helper mode, a router performs its helper function for a specific period of time. This time period is the lifetime value from the router that is in restart mode—minus the value of *lsage* in the received grace LSA. If the graceful restart succeeds in time, the helper's timer is stopped before it expires. If the helper's timer does expire, the adjacency to the restarting router is brought down, and normal OSPFv3 functionality resumes.
- The dead timer is not honored by the router that is in helper mode.
- A router in helper mode ceases to perform the helper function in any of the following cases:
 - The helper router is able to bring up a FULL adjacency with the restarting router.
 - The local timer for the helper function expires.

Protocol Shutdown Mode

In this mode the OSPFv3 operation is completely disabled. This is accomplished by flushing self-originated link state advertisements (LSAs), immediately bringing down local OSPFv3-supported interfaces, and clearing the Link State Database (LSDB). The non-local LSDB entries are removed by OSPFv3. These are not flooded (MaxAged).

The protocol shutdown mode can be invoked either manually through the **protocol shutdown** command that disables the protocol instance or when the OSPFv3 process runs out of memory. These events occur when protocol shut down is performed:

- The local Router LSA and all local Link LSAs are flushed. All other LSAs are eventually aged out by other OSPFv3 routers in the domain.
- OSPFv3 neighbors not yet in Full state with the local router are brought down with the Kill_Nbr event.
- After a three second delay, empty Hello packets are immediately sent to each neighbor that has an active adjacency.
 - An empty Hello packet is sent periodically until the dead_interval has elapsed.
 - When the dead_interval elapses, Hello packets are no longer sent.

After a Dead Hello interval delay (4 X Hello Interval), the following events are then performed:

- The LSA database from that OSPFv3 instance is cleared.
- All routes from RIB that were installed by OSPFv3 are purged.

The router will not respond to any OSPF control packets it receives from neighbors while in protocol shutdown state.

Protocol Restoration

The method of restoring the protocol is dependent on the trigger that originally invoked the shut down. If the OSPFv3 was shut down using the **protocol shutdown** command, then use the **no protocol shutdown** command to restore OSPFv3 back to normal operation. If the OSPFv3 was shutdown due to a Critical Memory message from the sysmon, then a Normal Memory message from sysmon, which indicates that sufficient memory has been restored to the processor, restores the OSPFv3 protocol to resume normal operation. When OSPFv3 is shutdown due to the Critical Memory trigger, it must be manually restarted when normal memory levels are restored on the route processor. It will not automatically restore itself.

These events occur when the OSPFv3 is restored:

1. All OSPFv3 interfaces are brought back up using the Hello packets and database exchange.
2. The local router and link LSAs are rebuilt and advertised.
3. The router replies normally to all OSPFv3 control messages received from neighbors.
4. Routes learned from other OSPFv3 routers are installed in RIB.

Graceful Restart Requirements and Restrictions

The requirements for supporting the Graceful Restart feature include:

- Cooperation of a router's neighbors during a graceful restart. In relation to the router on which OSPFv3 is restarting, each router is called a *helper*.
- All neighbors of the router that does a graceful restart must be capable of doing a graceful restart.
- A graceful restart does not occur upon the first-time startup of a router.
- OSPFv3 neighbor information and database information are not check-pointed.
- An OSPFv3 process rebuilds adjacencies after it restarts.

- To ensure consistent databases after a restart, the OSPFv3 configuration must be identical to the configuration before the restart. (This requirement applies to self-originated information in the local database.) A graceful restart can fail if configurations change during the operation. In this case, data forwarding would be affected. OSPFv3 resumes operation by regenerating all its LSAs and resynchronizing its database with all its neighbors.
- Although IPv6 FIB tables remain unchanged during a graceful restart, these tables eventually mark the routes as stale through the use of a holddown timer. Enough time is allowed for the protocols to rebuild state information and converge.
- The router on which OSPFv3 is restarting must send OSPFv3 hellos within the dead interval of the process restart. Protocols must be able to retain adjacencies with neighbors before the adjacency dead timer expires. The default for the dead timer is 40 seconds. If hellos do not arrive on the adjacency before the dead timer expires, the router takes down the adjacency. The OSPFv3 Graceful Restart feature does not function properly if the dead timer is configured to be less than the time required to send hellos after the OSPFv3 process restarts.
- Simultaneous graceful restart sessions on multiple routers are not supported on a single network segment. If a router determines that multiple routers are in restart mode, it terminates any local graceful restart operation.
- This feature utilizes the available support for changing the purge time of existing OSPFv3 routes in the Routing Information Base (RIB). When graceful restart is enabled, the purge timer is set to 90 seconds by default. If graceful restart is disabled, the purge timer setting is 0.
- This feature has an associated *grace* LSA. This link-scope LSA is type 11.
- According to the RFC, the OSPFv3 process should flush all old, self-originated LSAs during a restart. With the Graceful Restart feature, however, the router delays this flushing of unknown self-originated LSAs during a graceful restart. OSPFv3 can learn new information and build new LSAs to replace the old LSAs. When the delay is over, all old LSAs are flushed.
- If graceful restart is enabled, the adjacency creation time of all the neighbors is saved in the system database (SysDB). The purpose for saving the creation time is so that OSPFv3 can use the original adjacency creation time to display the uptime for that neighbor after the restart.

Warm Standby and Nonstop Routing for OSPF Version 2

OSPFv2 warm standby provides high availability across RP switchovers. With warm standby extensions, each process running on the active RP has a corresponding standby process started on the standby RP. A standby OSPF process can send and receive OSPF packets with no performance impact to the active OSPF process.

Nonstop routing (NSR) allows an RP failover, process restart, or in-service upgrade to be invisible to peer routers and ensures that there is minimal performance or processing impact. Routing protocol interactions between routers are not impacted by NSR. NSR is built on the warm standby extensions. NSR alleviates the requirement for Cisco NSF and IETF graceful restart protocol extensions.



Note It is recommended to set the hello timer interval to the default of 10 seconds. OSPF sessions may flap during switchover if hello-interval timer configured is less than default value.

Warm Standby and Nonstop Routing for OSPF Version 3

This feature helps OSPFv3 to initialize itself prior to Fail over (FO) and be ready to function before the failure occurs. It reduces the downtime during switchover. By default, the router sends hello packets every 40 seconds.

With warm standby process for each OSPF process running on the Active Route Processor, the corresponding OSPF process must start on the Standby RP. There are no changes in configuration for this feature.

Warm-Standby is always enabled. This is an advantage for the systems running OSPFv3 as their IGP when they do RP failover.

Multicast-Intact Support for OSPF

The multicast-intact feature provides the ability to run multicast routing (PIM) when IGP shortcuts are configured and active on the router. Both OSPFv2 and IS-IS support the multicast-intact feature.

You can enable multicast-intact in the IGP when multicast routing protocols (PIM) are configured and IGP shortcuts are configured on the router. IGP shortcuts are MPLS tunnels that are exposed to IGP. The IGP routes IP traffic over these tunnels to destinations that are downstream from the egress router of the tunnel (from an SPF perspective). PIM cannot use IGP shortcuts for propagating PIM joins, because reverse path forwarding (RPF) cannot work across a unidirectional tunnel.

When you enable multicast-intact on an IGP, the IGP publishes a parallel or alternate set of equal-cost next hops for use by PIM. These next hops are called *mcast-intact* next hops. The mcast-intact next hops have the following attributes:

- They are guaranteed not to contain any IGP shortcuts.
- They are not used for unicast routing but are used only by PIM to look up an IPv4 next-hop to a PIM source.
- They are not published to the FIB.
- When multicast-intact is enabled on an IGP, all IPv4 destinations that were learned through link-state advertisements are published with a set equal-cost mcast-intact next hops to the RIB. This attribute applies even when the native next hops have no IGP shortcuts.

In OSPF, the max-paths (number of equal-cost next hops) limit is applied separately to the native and mcast-intact next hops. The number of equal cost mcast-intact next hops is the same as that configured for the native next hops.

Load Balancing in OSPF Version 2 and OSPFv3

When a router learns multiple routes to a specific network by using multiple routing processes (or routing protocols), it installs the route with the lowest administrative distance in the routing table. Sometimes the router must select a route from among many learned by using the same routing process with the same administrative distance. In this case, the router chooses the path with the lowest cost (or metric) to the destination. Each routing process calculates its cost differently; the costs may need to be manipulated to achieve load balancing.

OSPF performs load balancing automatically. If OSPF finds that it can reach a destination through more than one interface and each path has the same cost, it installs each path in the routing table. The only restriction on the number of paths to the same destination is controlled by the **maximum-paths** (OSPF) command.

The range for maximum paths is from 1 to 8 and the default number of maximum paths is 8.

Configure Prefix Suppression for OSPF

Transit-only networks that connect two routers are usually configured with routing IP addresses that are advertised in the Links State Advertisements (LSAs). However, these prefixes are not needed for data traffic. Suppressing these prefixes would reduce the number of links in LSAs, thereby improving convergence and also reducing the vulnerability of potential remote attacks.

Prefixes can be suppressed for an OSPF process, an OSPF area, or for specific interfaces of a router.

Configure Prefix Suppression for a Router Running OSPF

Use the procedure in this section to configure prefix suppression for an OSPF process on a router.



Note

- If you suppress prefixes for an OSPF process on a router, the suppression is valid for all interfaces and areas associated with the router.
- When prefix suppression is configured on an NSSA ASBR, all interfaces on the routers have their prefixes suppressed, and the Type 7 LSAs have a forwarding address of 0. This stops the translation of Type 7 LSAs to Type 5 by the NSSA ABR. The workaround for this is to configure at least one loopback interface in the NSSA area, or one interface with prefix suppression disabled, so that the interface address is selected as the forwarding address for all the Type 7 LSAs.

1. Enter the global configuration mode and configure the interfaces of the router.

```
RP/0/# configure
RP/0/(config)# interface GigabitEthernet 0/0/0/1
RP/0/(config-if)# ipv4 address 10.1.1.1 255.255.255.0
RP/0/(config-if)# no shut
RP/0/(config-if)# exit
RP/0/(config)# interface Loopback 0
RP/0/(config-if)# ipv4 address 10.10.10.10 255.255.255.255
RP/0/(config-if)# no shut
RP/0/(config-if)# exit
```

2. Configure the OSPF process with prefix suppression.

```
RP/0/# configure
RP/0/(config)# router ospf pfx
RP/0/(config-ospf)# router-id 10.10.10.10
RP/0/(config-ospf)# prefix-suppression
```

3. Add the configured interfaces to the OSPF area.

```
RP/0/(config-ospf)# area 0
RP/0/(config-ospf-ar)# interface Loopback 0
RP/0/(config-ospf-ar-if)# exit
RP/0/(config-ospf-ar)# interface GigabitEthernet 0/0/0/1
RP/0/(config-ospf-ar-if)# network point-to-point
```

4. Exit the OSPF area configuration mode and commit your configuration.

```
RP/0/(config-ospf-ar-if)# exit
RP/0/(config-ospf-ar)# exit
RP/0/(config-ospf)# exit
RP/0/(config)# commit
RP/0/(config)# exit
```

5. Confirm your configuration.

```
RP/0/# show running-configuration
...
interface Loopback0
  ipv4 address 10.10.10.10 255.255.255.255
!
interface GigabitEthernet0/0/0/1
  ipv4 address 10.1.1.1 255.255.255.0
!
router ospf pfx
  router-id 10.10.10.10
  prefix-suppression
  area 0
    interface GigabitEthernet0/0/0/1
      network point-to-point
    !
  !
!
```

6. Verify whether prefix suppression is enabled.

```
RP/0/# show ospf interface
Fri Jun 17 15:13:08.470 IST

Interfaces for OSPF 1

GigabitEthernet0/0/0/1 is up, line protocol is up
  Internet Address 10.1.1.1/24, Area 0
  Process ID 1, Router ID 10.10.10.10, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State BDR, Priority 1, MTU 1500, MaxPktSz 1500
  Designated Router (ID) 10.10.10.20, Interface address 10.1.1.2
  Backup Designated router (ID) 10.10.10.30, Interface address 10.1.1.3
  Primary addresses not advertised
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:06:898
  Index 2/2, flood queue length 0
  Next 0(0)/0(0)
  Last flood scan length is 2, maximum is 2
  Last flood scan time is 0 msec, maximum is 0 msec
  LS Ack List: current length 0, high water mark 2
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 10.10.10.30 (Designated Router)
  Suppress hello for 0 neighbor(s)
  Multi-area interface Count is 0
```

If your output verifies that primary addresses are not advertised, then you have successfully configured prefix suppression for the OSPF process on the router.

Configure Prefix Suppression for an OSPF Area

Use the procedure in this section to configure prefix suppression for an OSPF area.



Note If you suppress prefixes on an area, the suppression is valid for all interfaces associated with the area.

1. Enter the global configuration mode and configure the interfaces of the router.

```
RP/0/# configure
RP/0/(config)# interface GigabitEthernet 0/0/0/1
RP/0/(config-if)# ipv4 address 10.1.1.1 255.255.255.0
```



```
RP/0/(config-if)# no shut
RP/0/(config-if)# exit
RP/0/(config)# interface Loopback 0
RP/0/(config-if)# ipv4 address 10.10.10.10 255.255.255.255
RP/0/(config-if)# no shut
RP/0/(config-if)# exit
```

2. Configure the OSPF area with prefix suppression.

```
RP/0/# configure
RP/0/(config)# router ospf pfx
RP/0/(config-ospf)# router-id 10.10.10.10
RP/0/(config-ospf)# area 0
RP/0/(config-ospf-ar)# prefix-suppression
```

3. Add the configured interfaces to the OSPF area.

```
RP/0/(config-ospf-ar)# interface Loopback 0
RP/0/(config-ospf-ar-if)# exit
RP/0/(config-ospf-ar)# interface GigabitEthernet 0/0/0/1
RP/0/(config-ospf-ar-if)# network point-to-point
```

4. Exit the OSPF area configuration mode and commit your configuration.

```
RP/0/(config-ospf-ar-if)# exit
RP/0/(config-ospf-ar)# exit
RP/0/(config-ospf)# exit
RP/0/(config)# commit
RP/0/(config)# exit
```

5. Confirm your configuration.

```
RP/0/# show running-configuration
...
interface Loopback0
  ipv4 address 10.10.10.10 255.255.255.255
!
interface GigabitEthernet0/0/0/1
  ipv4 address 10.1.1.1 255.255.255.0
!
router ospf pfx
  router-id 10.10.10.10
  area 0
    prefix-suppression
    interface GigabitEthernet0/0/0/1
      network point-to-point
    !
  !
!
```

6. Verify if prefix suppression is enabled.

```
RP/0/# show ospf interface
Fri Jun 17 15:13:08.470 IST

Interfaces for OSPF 1

GigabitEthernet0/0/0/1 is up, line protocol is up
  Internet Address 10.1.1.1/24, Area 0
  Process ID 1, Router ID 10.10.10.10, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State BDR, Priority 1, MTU 1500, MaxPktSz 1500
  Designated Router (ID) 10.10.10.20, Interface address 10.1.1.2
  Backup Designated router (ID) 10.10.10.30, Interface address 10.1.1.3
  Primary addresses not advertised
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
```

```

Hello due in 00:00:06:898
Index 2/2, flood queue length 0
Next 0(0)/0(0)
Last flood scan length is 2, maximum is 2
Last flood scan time is 0 msec, maximum is 0 msec
LS Ack List: current length 0, high water mark 2
Neighbor Count is 1, Adjacent neighbor count is 1
  Adjacent with neighbor 10.10.10.30 (Designated Router)
Suppress hello for 0 neighbor(s)
Multi-area interface Count is 0

```

If your output verifies that primary addresses are not advertised, then you have successfully configured prefix suppression for the OSPF area.

Configure Prefix Suppression for an OSPF Interface

Use the procedure in this section to configure prefix suppression for an OSPF interface.



Note If you suppress prefixes on an interface, suppression is valid only on that interface, and all other interfaces must be configured separately with prefix suppression.

1. Enter the global configuration mode and configure the interfaces of the router.

```

RP/0/# configure
RP/0/(config)# interface GigabitEthernet 0/0/0/1
RP/0/(config-if)# ipv4 address 10.1.1.1 255.255.255.0
RP/0/(config-if)# no shut
RP/0/(config-if)# exit
RP/0/(config)# interface Loopback 0
RP/0/(config-if)# ipv4 address 10.10.10.10 255.255.255.255
RP/0/(config-if)# no shut
RP/0/(config-if)# exit

```

2. Configure the OSPF area.

```

RP/0/# configure
RP/0/(config)# router ospf pfx
RP/0/(config-ospf)# router-id 10.10.10.10
RP/0/(config-ospf)# area 0

```

3. Add the configured interfaces to the OSPF area, and configure prefix suppression on the required interface.

```

RP/0/(config-ospf-ar)# interface Loopback 0
RP/0/(config-ospf-ar-if)# exit
RP/0/(config-ospf-ar)# interface GigabitEthernet 0/0/0/1
RP/0/(config-ospf-ar-if)# network point-to-point
RP/0/(config-ospf-ar-if)# prefix-suppression

```

4. Exit the OSPF area configuration mode and commit your configuration.

```

RP/0/(config-ospf-ar-if)# exit
RP/0/(config-ospf-ar)# exit
RP/0/(config-ospf)# exit
RP/0/(config)# commit
RP/0/(config)# exit

```

5. Confirm your configuration.

```

RP/0/# show running-configuration
...

```

```

interface Loopback0
  ipv4 address 10.10.10.10 255.255.255.255
!
interface GigabitEthernet0/0/0/1
  ipv4 address 10.1.1.1 255.255.255.0
!
router ospf pfx
  router-id 10.10.10.10
  area 0
    interface GigabitEthernet0/0/0/1
      network point-to-point
      prefix-suppression
    !
  !
!

```

6. Verify if prefix suppression is enabled.

```

RP/0/# show ospf interface
Fri Jun 17 15:13:08.470 IST

Interfaces for OSPF 1

GigabitEthernet0/0/0/1 is up, line protocol is up
  Internet Address 10.1.1.1/24, Area 0
  Process ID 1, Router ID 10.10.10.10, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State BDR, Priority 1, MTU 1500, MaxPktSz 1500
  Designated Router (ID) 10.10.10.20, Interface address 10.1.1.2
  Backup Designated router (ID) 10.10.10.30, Interface address 10.1.1.3
  Primary addresses not advertised
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:06:898
  Index 2/2, flood queue length 0
  Next 0(0)/0(0)
  Last flood scan length is 2, maximum is 2
  Last flood scan time is 0 msec, maximum is 0 msec
  LS Ack List: current length 0, high water mark 2
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 10.10.10.30 (Designated Router)
  Suppress hello for 0 neighbor(s)
  Multi-area interface Count is 0

```

If your output verifies that primary addresses are not advertised, then you have successfully configured prefix suppression on the interface.

Configure Prefix Suppression for OSPFv3

Transit-only networks that connect two routers are usually configured with routing IP addresses that are advertised in the Links State Advertisements (LSAs). However, these prefixes are not needed for data traffic. Suppressing these prefixes would reduce the number of links in LSAs, thereby improving convergence and also reducing the vulnerability of potential remote attacks.

Prefixes can be suppressed for an OSPF process, an OSPF area, or for specific interfaces of a router.

Configure Prefix Suppression for a Router Running OSPFv3

Use the procedure in this section to configure prefix suppression for an OSPFv3 process on a router.

**Note**

- If you suppress prefixes for an OSPF process on a router, the suppression is valid for all interfaces and areas associated with the router.
- When prefix suppression is configured on an NSSA ASBR, all interfaces on the routers have their prefixes suppressed, and the Type 7 LSAs have a forwarding address of 0. This stops the translation of Type 7 LSAs to Type 5 by the NSSA ABR. The workaround for this is to configure at least one loopback interface in the NSSA area, or one interface with prefix suppression disabled, so that the interface address is selected as the forwarding address for all the Type 7 LSAs.

1. Enter the global configuration mode and configure the interfaces of the router.

```
RP/0/# configure
RP/0/ (config) # interface GigabitEthernet 0/0/0/1
RP/0/ (config-if) # ipv6 address 2008:DB8::1/64
RP/0/ (config-if) # no shut
RP/0/ (config-if) # exit
RP/0/ (config) # interface Loopback 0
RP/0/ (config-if) # ipv6 address ::1/128
RP/0/ (config-if) # no shut
RP/0/ (config-if) # exit
```

2. Configure the OSPFv3 process with prefix suppression.

```
RP/0/# configure
RP/0/ (config) # router ospfv3 pfx
RP/0/ (config-ospfv3) # router-id 10.10.10.10
RP/0/ (config-ospfv3) # prefix-suppression
```

3. Add the configured interfaces to the OSPFv3 area.

```
RP/0/ (config-ospfv3) # area 0
RP/0/ (config-ospfv3-ar) # interface Loopback 0
RP/0/ (config-ospfv3-ar-if) # exit
RP/0/ (config-ospfv3-ar) # interface GigabitEthernet 0/0/0/1
RP/0/ (config-ospfv3-ar-if) # network point-to-point
```

4. Exit the OSPFv3 area configuration mode and commit your configuration.

```
RP/0/ (config-ospfv3-ar-if) # exit
RP/0/ (config-ospfv3-ar) # exit
RP/0/ (config-ospfv3) # exit
RP/0/ (config) # commit
RP/0/ (config) # exit
```

5. Confirm your configuration.

```
RP/0/# show running-configuration
...
interface Loopback0
  ipv6 address ::1/128
!
interface GigabitEthernet0/0/0/1
  ipv6 address 2008:DB8::1/64
!
router ospfv3 pfx
  router-id 10.10.10.10
  prefix-suppression
  area 0
    interface GigabitEthernet0/0/0/1
      network point-to-point
```

```
!
!
!
```

6. Verify whether prefix suppression is enabled.

```
RP/0/# show ospfv3 interface
Fri Jun 17 15:13:08.470 IST
GigabitEthernet0/0/0/2 is up, line protocol is up
  Link Local address fe80::45:caff:feaf:72af, Interface ID 5
  Area 0, Process ID pfx, Instance ID 0, Router ID 10.10.10.10
  Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State DOWN, Priority 1
  No designated router on this network
  No backup designated router on this network
Addresses not advertised
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
  Index 0/1/1, flood queue length 0
  Next 0(0)/0(0)/0(0)
  Last flood scan length is 0, maximum is 0
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 0, Adjacent neighbor count is 0
  Suppress hello for 0 neighbor(s)
  Reference count is 0
```

If your output verifies that primary addresses are not advertised, then you have successfully configured prefix suppression for the OSPFv3 process on the router.

Configure Prefix Suppression for an OSPFv3 Area

Use the procedure in this section to configure prefix suppression for an OSPFv3 area.



Note If you suppress prefixes on an area, the suppression is valid for all interfaces associated with the area.

1. Enter the global configuration mode and configure the interfaces of the router.

```
RP/0/# configure
RP/0/(config)# interface GigabitEthernet 0/0/0/1
RP/0/(config-if)# ipv6 address 2008:DB8::1/64
RP/0/(config-if)# no shut
RP/0/(config-if)# exit
RP/0/(config)# interface Loopback 0
RP/0/(config-if)# ipv6 address ::1/128
RP/0/(config-if)# no shut
RP/0/(config-if)# exit
```

2. Configure the OSPFv3 area with prefix suppression.

```
RP/0/# configure
RP/0/(config)# router ospfv3 pfx
RP/0/(config-ospfv3)# router-id 10.10.10.10
RP/0/(config-ospfv3)# area 0
RP/0/(config-ospfv3-ar)# prefix-suppression
```

3. Add the configured interfaces to the OSPFv3 area.

```
RP/0/(config-ospfv3-ar)# interface Loopback 0
RP/0/(config-ospfv3-ar-if)# exit
RP/0/(config-ospfv3-ar)# interface GigabitEthernet 0/0/0/1
RP/0/(config-ospfv3-ar-if)# network point-to-point
```

- Exit the OSPFv3 area configuration mode and commit your configuration.

```
RP/0/ (config-ospfv3-ar-if)# exit
RP/0/ (config-ospfv3-ar)# exit
RP/0/ (config-ospfv3)# exit
RP/0/ (config)# commit
RP/0/ (config)# exit
```

- Confirm your configuration.

```
RP/0/# show running-configuration
...
interface Loopback0
  ipv6 address ::1/128
!
interface GigabitEthernet0/0/0/1
  ipv6 address 2008:DB8::1/64
!
router ospfv3 pfx
  router-id 10.10.10.10
  area 0
    prefix-suppression
      interface GigabitEthernet0/0/0/1
        network point-to-point
      !
    !
  !
```

- Verify if prefix suppression is enabled.

```
RP/0/# show ospfv3 interface
Fri Jun 17 15:13:08.470 IST
GigabitEthernet0/0/0/2 is up, line protocol is up
  Link Local address fe80::45:caff:feaf:72af, Interface ID 5
  Area 0, Process ID pfx, Instance ID 0, Router ID 10.10.10.10
  Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State DOWN, Priority 1
  No designated router on this network
  No backup designated router on this network
  Addresses not advertised
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
  Index 0/1/1, flood queue length 0
  Next 0(0)/0(0)/0(0)
  Last flood scan length is 0, maximum is 0
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 0, Adjacent neighbor count is 0
  Suppress hello for 0 neighbor(s)
  Reference count is 0
```

If your output verifies that primary addresses are not advertised, then you have successfully configured prefix suppression for the OSPFv3 area.

Configure Prefix Suppression for an OSPFv3 Interface

Use the procedure in this section to configure prefix suppression for an OSPFv3 interface.



Note If you suppress prefixes on an interface, suppression is valid only on that interface, and all other interfaces must be configured separately with prefix suppression.

- Enter the global configuration mode and configure the interfaces of the router.

```
RP/0/# configure
RP/0/(config)# interface GigabitEthernet 0/0/0/1
RP/0/(config-if)# ipv6 address 2008:DB8::1/64
RP/0/(config-if)# no shut
RP/0/(config-if)# exit
RP/0/(config)# interface loopback 0
RP/0/(config-if)# ipv6 address ::1/128
RP/0/(config-if)# no shut
RP/0/(config-if)# exit
```

2. Configure the OSPFv3 area.

```
RP/0/# configure
RP/0/(config)# router ospfv3 pfx
RP/0/(config-ospfv3)# router-id 10.10.10.10
RP/0/(config-ospfv3)# area 0
```

3. Add the configured interfaces to the OSPFv3 area, and configure prefix suppression on the required interface.

```
RP/0/(config-ospfv3-ar)# interface Loopback 0
RP/0/(config-ospfv3-ar-if)# exit
RP/0/(config-ospfv3-ar)# interface GigabitEthernet 0/0/0/1
RP/0/(config-ospfv3-ar-if)# network point-to-point
RP/0/(config-ospfv3-ar-if)# prefix-suppression
```

4. Exit the OSPFv3 area configuration mode and commit your configuration.

```
RP/0/(config-ospfv3-ar-if)# exit
RP/0/(config-ospfv3-ar)# exit
RP/0/(config-ospfv3)# exit
RP/0/(config)# commit
RP/0/(config)# exit
```

5. Confirm your configuration.

```
RP/0/# show running-configuration
...
interface Loopback0
  ipv4 address 10.10.10.10 255.255.255.255
  !
interface GigabitEthernet0/0/0/1
  ipv4 address 10.1.1.1 255.255.255.0
  !
router ospfv3 pfx
  router-id 10.10.10.10
  area 0
    interface GigabitEthernet0/0/0/1
      network point-to-point
      prefix-suppression
    !
  !
  !
```

6. Verify if prefix suppression is enabled.

```
RP/0/# show ospfv3 interface
Fri Jun 17 15:13:08.470 IST
GigabitEthernet0/0/0/2 is up, line protocol is up
  Link Local address fe80::45:caff:feaf:72af, Interface ID 5
  Area 0, Process ID pfx, Instance ID 0, Router ID 10.10.10.10
  Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State DOWN, Priority 1
  No designated router on this network
  No backup designated router on this network
```

Addresses not advertised

```

Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
Index 0/1/1, flood queue length 0
Next 0(0)/0(0)/0(0)
Last flood scan length is 0, maximum is 0
Last flood scan time is 0 msec, maximum is 0 msec
Neighbor Count is 0, Adjacent neighbor count is 0
Suppress hello for 0 neighbor(s)
Reference count is 0

```

If your output verifies that primary addresses are not advertised, then you have successfully configured prefix suppression on the interface.

Multi-Area Adjacency for OSPF Version 2

The multi-area adjacency feature for OSPFv2 allows a link to be configured on the primary interface in more than one area so that the link could be considered as an intra-area link in those areas and configured as a preference over more expensive paths.

This feature establishes a point-to-point unnumbered link in an OSPF area. A point-to-point link provides a topological path for that area, and the primary adjacency uses the link to advertise the link consistent with draft-ietf-ospf-multi-area-adj-06.

The following are multi-area interface attributes and limitations:

- Exists as a logical construct over an existing primary interface for OSPF; however, the neighbor state on the primary interface is independent of the multi-area interface.
- Establishes a neighbor relationship with the corresponding multi-area interface on the neighboring router. A mixture of multi-area and primary interfaces is not supported.
- Advertises an unnumbered point-to-point link in the router link state advertisement (LSA) for the corresponding area when the neighbor state is full.
- Created as a point-to-point network type. You can configure multi-area adjacency on any interface where only two OSPF speakers are attached. In the case of native broadcast networks, the interface must be configured as an OSPF point-to-point type using the **network point-to-point** command to enable the interface for a multi-area adjacency.
- Inherits the Bidirectional Forwarding Detection (BFD) characteristics from its primary interface. BFD is not configurable under a multi-area interface; however, it is configurable under the primary interface.

The multi-area interface inherits the interface characteristics from its primary interface, but some interface characteristics can be configured under the multi-area interface configuration mode as shown below:

```

RP/0/(config-ospf-ar) # multi-area-interface GigabitEthernet 0/1/0/3
RP/0/(config-ospf-ar-mif) # ?
authentication          Enable authentication
authentication-key      Authentication password (key)
cost                    Interface cost
cost-fallback           Cost when cumulative bandwidth goes below the threshold
database-filter         Filter OSPF LSA during synchronization and flooding
dead-interval           Interval after which a neighbor is declared dead
distribute-list         Filter networks in routing updates
hello-interval          Time between HELLO packets
message-digest-key      Message digest authentication password (key)
mtu-ignore              Enable/Disable ignoring of MTU in DBD packets
packet-size             Customize size of OSPF packets upto MTU

```



```
retransmit-interval  Time between retransmitting lost link state advertisements
transmit-delay        Estimated time needed to send link-state update packet
```

```
RP/0/ (config-ospf-ar-mif) #
```

Label Distribution Protocol IGP Auto-configuration for OSPF

Label Distribution Protocol (LDP) Interior Gateway Protocol (IGP) auto-configuration simplifies the procedure to enable LDP on a set of interfaces used by an IGP instance, such as OSPF. LDP IGP auto-configuration can be used on a large number of interfaces (for example, when LDP is used for transport in the core) and on multiple OSPF instances simultaneously.

This feature supports the IPv4 unicast address family for the default VPN routing and forwarding (VRF) instance.

LDP IGP auto-configuration can also be explicitly disabled on an individual interface basis under LDP using the **ldp auto-config disable** command. This allows LDP to receive all OSPF interfaces minus the ones explicitly disabled.

See [for information on configuring LDP IGP auto-configuration.](#)

OSPF Authentication Message Digest Management

All OSPF routing protocol exchanges are authenticated and the method used can vary depending on how authentication is configured. When using cryptographic authentication, the OSPF routing protocol uses the Message Digest 5 (MD5) authentication algorithm to authenticate packets transmitted between neighbors in the network. For each OSPF protocol packet, a key is used to generate and verify a message digest that is appended to the end of the OSPF packet. The message digest is a one-way function of the OSPF protocol packet and the secret key. Each key is identified by the combination of interface used and the key identification. An interface may have multiple keys active at any time.

To manage the rollover of keys and enhance MD5 authentication for OSPF, you can configure a container of keys called a *keychain* with each key comprising the following attributes: generate/accept time, key identification, and authentication algorithm.

GTSM TTL Security Mechanism for OSPF

OSPF is a link state protocol that requires networking devices to detect topological changes in the network, flood Link State Advertisement (LSA) updates to neighbors, and quickly converge on a new view of the topology. However, during the act of receiving LSAs from neighbors, network attacks can occur, because there are no checks that unicast or multicast packets are originating from a neighbor that is one hop away or multiple hops away over virtual links.

For virtual links, OSPF packets travel multiple hops across the network; hence, the TTL value can be decremented several times. For these type of links, a minimum TTL value must be allowed and accepted for multiple-hop packets.

To filter network attacks originating from invalid sources traveling over multiple hops, the Generalized TTL Security Mechanism (GTSM), RFC 3682, is used to prevent the attacks. GTSM filters link-local addresses and allows for only one-hop neighbor adjacencies through the configuration of TTL value 255. The TTL value in the IP header is set to 255 when OSPF packets are originated, and checked on the received OSPF packets

against the default GTSM TTL value 255 or the user configured GTSM TTL value, blocking unauthorized OSPF packets originated from TTL hops away.

Path Computation Element for OSPFv2

A PCE is an entity (component, application, or network node) that is capable of computing a network path or route based on a network graph and applying computational constraints.

PCE is accomplished when a PCE address and client is configured for MPLS-TE. PCE communicates its PCE address and capabilities to OSPF then OSPF packages this information in the PCE Discovery type-length-value (TLV) (Type 2) and reoriginates the RI LSA. OSPF also includes the Router Capabilities TLV (Type 1) in all its RI LSAs. The PCE Discovery TLV contains the PCE address sub-TLV (Type 1) and the Path Scope Sub-TLV (Type 2).

The PCE Address Sub-TLV specifies the IP address that must be used to reach the PCE. It should be a loop-back address that is always reachable, this TLV is mandatory, and must be present within the PCE Discovery TLV. The Path Scope Sub-TLV indicates the PCE path computation scopes, which refers to the PCE ability to compute or participate in the computation of intra-area, inter-area, inter-AS or inter-layer TE LSPs.

PCE extensions to OSPFv2 include support for the Router Information Link State Advertisement (RI LSA). OSPFv2 is extended to receive all area scopes (LSA Types 9, 10, and 11). However, OSPFv2 originates only area scope Type 10.

For detailed information for the Path Computation Element feature see the *Implementing MPLS Traffic Engineering on* module of the and the following IETF drafts:

- draft-ietf-ospf-cap-09
- draft-ietf-pce-disco-ospf-00

OSPF IP Fast Reroute Loop Free Alternate

The OSPF IP Fast Reroute (FRR) Loop Free Alternate (LFA) computation supports these:

- Fast rerouting capability by using IP forwarding and routing
- Handles failure in the line cards in minimum time

Management Information Base (MIB) for OSPFv3

Cisco IOS XR supports full MIBs and traps for OSPFv3, as defined in RFC 5643. The RFC 5643 defines objects of the Management Information Base (MIB) for use with the Open Shortest Path First (OSPF) Routing Protocol for IPv6 (OSPF version 3).

The OSPFv3 MIB implementation is based on the IETF draft *Management Information Base for OSPFv3 (draft-ietf-ospf-ospfv3-mib-8)*. Users need to update the NMS application to pick up the new MIB when upgraded to RFC 5643.

Refer to the *Cisco ASR 9000 Series Aggregation Services Router MIB Specification Guide* for more information on Cisco IOS XR MIB support.

Multiple OSPFv3 Instances

SNMPv3 supports "contexts" that can be used to implement MIB views on multiple OSPFv3 instances, in the same system.

Management Information Base (MIB) for OSPFv2

Cisco IOS XR supports full MIBs and traps for OSPFv2, as defined in RFC 4750. The RFC 4750 defines objects of the Management Information Base (MIB) for use with the Open Shortest Path First (OSPF) Routing Protocol for IPv4 (OSPF version 2).

The OSPFv2 MIB implementation is based on the IETF draft Management Information Base for OSPFv2 ([draft-ietf-ospf-mib \(ospf WG\)](#)).

To know more about MIBS, use the [MIB Locator](#).

VRF-lite Support for OSPFv2

VRF-lite capability is enabled for OSPF version 2 (OSPFv2). VRF-lite is the virtual routing and forwarding (VRF) deployment without the BGP/MPLS based backbone. In VRF-lite, individual provider edge (PE) routers are directly connected using VRF interfaces. To enable VRF-lite in OSPFv2, configure the **capability vrf-lite** command in VRF configuration mode. When VRF-lite is configured, the DN bit processing and the automatic Area Border Router (ABR) status setting are disabled.

OSPFv3 Timers Update

The Open Shortest Path First version 3 (OSPFv3) timers link-state advertisements (LSAs) and shortest path first (SPF) throttle default values are updated to:

- **timers throttle lsa all**—*start-interval*: 50 milliseconds and *hold-interval*: 200 milliseconds
- **timers throttle spf**—*spf-start*: 50 milliseconds, *spf-hold*: 200 milliseconds, *spf-max-wait*: 5000 milliseconds

Unequal Cost Multipath Load-balancing for OSPF

The unequal cost multipath (UCMP) load-balancing adds the capability with Open Shortest Path First (OSPF) to load-balance traffic proportionally across multiple paths, with different cost. Without UCMP enabled, only the best cost paths are discovered by OSPF (ECMP) and alternate higher cost paths are not computed.

Generally, higher bandwidth links have lower IGP metrics configured, so that they form the shortest IGP paths. With the UCMP load-balancing enabled, IGP can use even lower bandwidth links or higher cost links for traffic, and can install these paths to the forwarding information base (FIB). OSPF installs multiple paths to the same destination in FIB, but each path will have a 'load metric/weight' associated with it. FIB uses this load metric/weight to decide the amount of traffic that needs to be sent on a higher bandwidth path and the amount of traffic that needs to be sent on a lower bandwidth path.

The UCMP computation is provided under OSPF VRF context, enabling UCMP computation for a particular VRF. For default VRF the configuration is done under the OSPF global mode. The UCMP configuration is also provided with a prefix-list option, which would limit the UCMP computation only for the prefixes present in the prefix-list. If prefix-list option is not provided, UCMP computation is done for the reachable prefixes in OSPF. The number of UCMP paths to be considered and installed is controlled using the **variance**

configuration. Variance value identifies the range for the UCMP path metric to be considered for installation into routing information base (RIB/FIB) and is defined in terms of a percentage of the primary path metric. Total number of paths, including ECMP and UCMP paths together is limited by the max-path configuration or by the max-path capability of the platform.

There is an option to exclude an interface from being used for UCMP computation. If it is desired that a particular interface should not be considered as a UCMP nexthop, for any prefix, then use the UCMP **exclude interface** command to configure the interface to be excluded from UCMP computation.

Enabling the UCMP configuration indicates that OSPF should perform UCMP computation for the all the reachable OSPF prefixes or all the prefixes permitted by the prefix-list, if the prefix-list option is used. The UCMP computation happens only after the primary SPF and route calculation is completed. There would be a configurable delay (default delay is 100 ms) from the time primary route calculation is completed and UCMP computation is started. Use the UCMP **delay-interval** command to configure the delay between primary SPF completion and start of UCMP computation. UCMP computation will be done during the fast re-route computation (IPFRR does not need to be enabled for UCMP computation to be performed). If IPFRR is enabled, the fast re-route backup paths will be calculated for both the primary equal cost multipath (ECMP) paths and the UCMP paths.

To manually adjust UCMP ratio, use any command that changes the metric of the link.

- By using the bandwidth command in interface configuration mode
- By adjusting the OSPF interface cost on the link

How to Implement OSPF

This section contains the following procedures:

Enable OSPF

This task explains how to perform the minimum OSPF configuration on your router that is to enable an OSPF process with a router ID, configure a backbone or nonbackbone area, and then assign one or more interfaces on which OSPF runs.

Before you begin

Although you can configure OSPF before you configure an IP address, no OSPF routing occurs until at least one IP address is configured.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **area** *area-id*
5. **interface** *type interface-path-id*
6. Repeat Step 5 for each interface that uses OSPF.

7. **log adjacency changes** [**detail**] [**enable** | **disable**]
8. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
Enters mode.
```

Step 2 Do one of the following:

- **router ospf** *process-name*
- **router ospfv3** *process-name*

Example:

```
RP/0/(config)# router ospf 1
or
```

```
RP/0/(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }

Example:

```
RP/0/(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IP address as the router ID.

Step 4 **area** *area-id*

Example:

```
RP/0/(config-ospf)# area 0
```

Enters area configuration mode and configures an area for the OSPF process.

- Backbone areas have an area ID of 0.
- Nonbackbone areas have a nonzero area ID.
- The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 5 `interface type interface-path-id`**Example:**

```
RP/0/(config-ospf-ar)# interface TenGigE 0/1/0/3
```

Enters interface configuration mode and associates one or more interfaces for the area configured in Step 4.

Step 6 Repeat Step 5 for each interface that uses OSPF.

—

Step 7 `log adjacency changes [detail] [enable | disable]`**Example:**

```
RP/0/(config-ospf-ar-if)# log adjacency changes detail
```

(Optional) Requests notification of neighbor changes.

- By default, this feature is enabled.
- The messages generated by neighbor changes are considered notifications, which are categorized as severity Level 5 in the **logging console** command. The **logging console** command controls which severity level of messages are sent to the console. By default, all severity level messages are sent.

Step 8 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Enable OSPF: Example

OSPF areas must be explicitly configured, and interfaces configured under the area configuration mode are explicitly bound to that area. In this example, interface 10.1.2.0/24 is bound to area 0 and interface 10.1.3.0/24 is bound to area 1.

```
interface TenGigE 0/3/0/0
 ip address 10.1.2.1 255.255.255.0
 negotiation auto
!
interface TenGigE 0/3/0/1
 ip address 10.1.3.1 255.255.255.0
 negotiation auto
!
router ospf 1
 router-id 10.2.3.4
 area 0
 interface TenGigE 0/3/0/0
!
```

```

area 1
 interface TenGigE 0/3/0/1
 !
 !

```

Configuring Stub and Not-So-Stubby Area Types

This task explains how to configure the stub area and the NSSA for OSPF.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **area** *area-id*
5. Do one of the following:
 - **stub** [**no-summary**]
 - **nssa** [**no-redistribution**] [**default-information-originate**] [**no-summary**] [**translate**] [**translate always**]
6. Do one of the following:
 - **stub**
 - **nssa**
7. **default-cost** *cost*
8. Use the **commit** or **end** command.
9. Repeat this task on all other routers in the stub area or NSSA.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	Do one of the following: <ul style="list-style-type: none"> • router ospf <i>process-name</i> • router ospfv3 <i>process-name</i> Example: RP/0/(config)# router ospf 1 or	Enables OSPF routing for the specified routing process and places the router in router configuration mode. or Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode. Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.

	Command or Action	Purpose
	RP/0/(config)# router ospfv3 1	
Step 3	router-id { <i>router-id</i> } Example: RP/0/(config-ospf)# router-id 192.168.4.3	Configures a router ID for the OSPF process. Note We recommend using a stable IP address as the router ID.
Step 4	area <i>area-id</i> Example: RP/0/(config-ospf)# area 1	Enters area configuration mode and configures a nonbackbone area for the OSPF process. <ul style="list-style-type: none"> The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.
Step 5	Do one of the following: <ul style="list-style-type: none"> stub [no-summary] nssa [no-redistribution] [default-information-originate] [no-summary] [translate] [translate always] Example: RP/0/(config-ospf-ar)# stub no summary or RP/0/(config-ospf-ar)# nssa no-redistribution or RP/0/(config-ospf-ar)# nssa translate <type number> always	Defines the nonbackbone area as a stub area. <ul style="list-style-type: none"> Specify the no-summary keyword to further reduce the number of LSAs sent into a stub area. This keyword prevents the ABR from sending summary link-state advertisements (Type 3) in the stub area. or Defines an area as an NSSA.
Step 6	Do one of the following: <ul style="list-style-type: none"> stub nssa Example: RP/0/(config-ospf-ar)# stub or RP/0/(config-ospf-ar)# nssa	(Optional) Turns off the options configured for stub and NSSA areas. <ul style="list-style-type: none"> If you configured the stub and NSSA areas using the optional keywords (no-summary , no-redistribution , default-information-originate , and translate) in Step 5, you must now reissue the stub and nssa commands without the keywords—rather than using the no form of the command. For example, the no nssa default-information-originate form of the command changes the NSSA area into a normal area that inadvertently brings down the existing adjacencies in that area.

	Command or Action	Purpose
Step 7	default-cost <i>cost</i> Example: RP/0/ (config-ospf-ar)#default-cost 15	(Optional) Specifies a cost for the default summary route sent into a stub area or an NSSA. <ul style="list-style-type: none"> • Use this command only on ABRs attached to the NSSA. Do not use it on any other routers in the area. • The default cost is 1.
Step 8	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.
Step 9	Repeat this task on all other routers in the stub area or NSSA.	—

Configuring Neighbors for Nonbroadcast Networks

This task explains how to configure neighbors for a nonbroadcast network. This task is optional.

Before you begin

Configuring NBMA networks as either broadcast or nonbroadcast assumes that there are virtual circuits from every router to every router or fully meshed network.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **area** *area-id*
5. **network** { **broadcast** | **non-broadcast** | { **point-to-multipoint** [**non-broadcast**] | **point-to-point** } }
6. **dead-interval** *seconds*
7. **hello-interval** *seconds*
8. **interface** *type interface-path-id*
9. Do one of the following:

- **neighbor** *ip-address* [**priority** *number*] [**poll-interval** *seconds*] [**cost** *number*]
- **neighbor** *ipv6-link-local-address* [**priority** *number*] [**poll-interval** *seconds*] [**cost** *number*] [**database-filter** [**all**]]

10. Repeat Step 9 for all neighbors on the interface.
11. **exit**
12. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# <code>configure</code>	Enters mode.
Step 2	Do one of the following: <ul style="list-style-type: none"> • router ospf <i>process-name</i> • router ospfv3 <i>process-name</i> Example: RP/0/(config)# <code>router ospf 1</code> or RP/0/(config)# <code>router ospfv3 1</code>	Enables OSPF routing for the specified routing process and places the router in router configuration mode. or Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode. Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.
Step 3	router-id { <i>router-id</i> } Example: RP/0/(config-ospf)# <code>router-id 192.168.4.3</code>	Configures a router ID for the OSPF process. Note We recommend using a stable IP address as the router ID.
Step 4	area <i>area-id</i> Example: RP/0/(config-ospf)# <code>area 0</code>	Enters area configuration mode and configures an area for the OSPF process. <ul style="list-style-type: none"> • The example configures a backbone area. • The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.
Step 5	network { broadcast non-broadcast { point-to-multipoint [non-broadcast] point-to-point } } Example: RP/0/(config-ospf-ar)# <code>network non-broadcast</code>	Configures the OSPF network type to a type other than the default for a given medium. <ul style="list-style-type: none"> • The example sets the network type to NBMA.

	Command or Action	Purpose
Step 6	<p>dead-interval <i>seconds</i></p> <p>Example:</p> <pre>RP/0/(config-ospf-ar)# dead-interval 40</pre>	(Optional) Sets the time to wait for a hello packet from a neighbor before declaring the neighbor down.
Step 7	<p>hello-interval <i>seconds</i></p> <p>Example:</p> <pre>RP/0/(config-ospf-ar)# hello-interval 10</pre>	<p>(Optional) Specifies the interval between hello packets that OSPF sends on the interface.</p> <p>Note It is recommended to set the hello timer interval to the default of 10 seconds. OSPF sessions may flap during switchover if hello-interval timer configured is less than default value.</p>
Step 8	<p>interface <i>type interface-path-id</i></p> <p>Example:</p> <pre>RP/0/(config-ospf-ar)# interface GigabitEthernet 0/2/0/0</pre>	<p>Enters interface configuration mode and associates one or more interfaces for the area configured in Step 4.</p> <ul style="list-style-type: none"> In this example, the interface inherits the nonbroadcast network type and the hello and dead intervals from the areas because the values are not set at the interface level.
Step 9	<p>Do one of the following:</p> <ul style="list-style-type: none"> neighbor <i>ip-address</i> [priority <i>number</i>] [poll-interval <i>seconds</i>] [cost <i>number</i>] neighbor <i>ipv6-link-local-address</i> [priority <i>number</i>] [poll-interval <i>seconds</i>] [cost <i>number</i>] [database-filter [all]] <p>Example:</p> <pre>RP/0/(config-ospf-ar-if)# neighbor 10.20.20.1 priority 3 poll-interval 15</pre> <p>or</p> <pre>RP/0/(config-ospf-ar-if)# neighbor fe80::3203:a0ff:fe9d:f3fe</pre>	<p>Configures the IPv4 address of OSPF neighbors interconnecting to nonbroadcast networks.</p> <p>or</p> <p>Configures the link-local IPv6 address of OSPFv3 neighbors.</p> <ul style="list-style-type: none"> The <i>ipv6-link-local-address</i> argument must be in the form documented in RFC 2373 in which the address is specified in hexadecimal using 16-bit values between colons. The priority keyword notifies the router that this neighbor is eligible to become a DR or BDR. The priority value should match the actual priority setting on the neighbor router. The neighbor priority default value is zero. This keyword does not apply to point-to-multipoint interfaces. The poll-interval keyword does not apply to point-to-multipoint interfaces. RFC 1247 recommends that this value be much larger than the hello interval. The default is 120 seconds (2 minutes). Neighbors with no specific cost configured assumes the cost of the interface, based on the cost command. On point-to-multipoint interfaces, cost number is the only keyword and argument combination that

	Command or Action	Purpose
		works. The cost keyword does not apply to NBMA networks. <ul style="list-style-type: none"> The database-filter keyword filters outgoing LSAs to an OSPF neighbor. If you specify the all keyword, incoming and outgoing LSAs are filtered. Use with extreme caution since filtering may cause the routing topology to be seen as entirely different between two neighbors, resulting in “black-holing” of data traffic or routing loops.
Step 10	Repeat Step 9 for all neighbors on the interface.	—
Step 11	exit Example: RP/0/(config-ospf-ar-if)# exit	Enters area configuration mode.
Step 12	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> Yes — Saves configuration changes and exits the configuration session. No —Exits the configuration session without committing the configuration changes. Cancel —Remains in the configuration session, without committing the configuration changes.

Configuring Authentication at Different Hierarchical Levels for OSPF Version 2

This task explains how to configure MD5 (secure) authentication on the OSPF router process, configure one area with plain text authentication, and then apply one interface with clear text (null) authentication.



Note Authentication configured at the interface level overrides authentication configured at the area level and the router process level. If an interface does not have authentication specifically configured, the interface inherits the authentication parameter value from a higher hierarchical level.

Before you begin

If you choose to configure authentication, you must first decide whether to configure plain text or MD5 authentication, and whether the authentication applies to all interfaces in a process, an entire area, or specific

interfaces. See [Route Authentication Methods for OSPF, on page 86](#) for information about each type of authentication and when you should use a specific method for your network.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **router-id** { *router-id* }
4. **authentication** [**message-digest** | **null**]
5. **message-digest-key** *key-id md5* { *key* | **clear** *key* | **encrypted** *key* | **LINE** }
6. **area** *area-id*
7. **interface** *type interface-path-id*
8. Repeat Step 7 for each interface that must communicate, using the same authentication.
9. **exit**
10. **area** *area-id*
11. **authentication** [**message-digest** | **null**]
12. **interface** *type interface-path-id*
13. Repeat Step 12 for each interface that must communicate, using the same authentication.
14. **interface** *type interface-path-id*
15. **authentication** [**message-digest** | **null**]
16. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	router ospf <i>process-name</i> Example: RP/0/(config)# router ospf 1	Enables OSPF routing for the specified routing process and places the router in router configuration mode. Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.
Step 3	router-id { <i>router-id</i> } Example: RP/0/(config-ospf)# router-id 192.168.4.3	Configures a router ID for the OSPF process.
Step 4	authentication [message-digest null] Example: RP/0/(config-ospf)#authentication message-digest	Enables MD5 authentication for the OSPF process. • This authentication type applies to the entire router process unless overridden by a lower hierarchical level such as the area or interface.

	Command or Action	Purpose
Step 5	<p>message-digest-key <i>key-id</i> md5 { <i>key</i> clear <i>key</i> encrypted <i>key</i> LINE }</p> <p>Example:</p> <pre>RP/0/(config-ospf)#message-digest-key 4 md5 yourkey</pre>	<p>Specifies the MD5 authentication key for the OSPF process.</p> <ul style="list-style-type: none"> The neighbor routers must have the same key identifier.
Step 6	<p>area <i>area-id</i></p> <p>Example:</p> <pre>RP/0/(config-ospf)# area 0</pre>	<p>Enters area configuration mode and configures a backbone area for the OSPF process.</p>
Step 7	<p>interface <i>type interface-path-id</i></p> <p>Example:</p> <pre>RP/0/(config-ospf-ar)# interface GigabitEthernet 0/1/0/3</pre>	<p>Enters interface configuration mode and associates one or more interfaces to the backbone area.</p> <ul style="list-style-type: none"> All interfaces inherit the authentication parameter values specified for the OSPF process (Step 4, Step 5, and Step 6).
Step 8	<p>Repeat Step 7 for each interface that must communicate, using the same authentication.</p>	—
Step 9	<p>exit</p> <p>Example:</p> <pre>RP/0/(config-ospf-ar)# exit</pre>	<p>Enters area OSPF configuration mode.</p>
Step 10	<p>area <i>area-id</i></p> <p>Example:</p> <pre>RP/0/(config-ospf)# area 1</pre>	<p>Enters area configuration mode and configures a nonbackbone area 1 for the OSPF process.</p> <ul style="list-style-type: none"> The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.
Step 11	<p>authentication [message-digest null]</p> <p>Example:</p> <pre>RP/0/(config-ospf-ar)# authentication</pre>	<p>Enables Type 1 (plain text) authentication that provides no security.</p> <ul style="list-style-type: none"> The example specifies plain text authentication (by not specifying a keyword). Use the authentication-key command in interface configuration mode to specify the plain text password.
Step 12	<p>interface <i>type interface-path-id</i></p> <p>Example:</p> <pre>RP/0/(config-ospf-ar)# interface GigabitEthernet 0/1/0/0</pre>	<p>Enters interface configuration mode and associates one or more interfaces to the nonbackbone area 1 specified in Step 7.</p> <ul style="list-style-type: none"> All interfaces configured inherit the authentication parameter values configured for area 1.

	Command or Action	Purpose
Step 13	Repeat Step 12 for each interface that must communicate, using the same authentication.	—
Step 14	interface <i>type interface-path-id</i> Example: <pre>RP/0/(config-ospf-ar)# interface GigabitEthernet 0/3/0/0</pre>	Enters interface configuration mode and associates one or more interfaces to a different authentication type.
Step 15	authentication [message-digest null] Example: <pre>RP/0/(config-ospf-ar-if)# authentication null</pre>	Specifies no authentication on GigabitEthernet interface 0/3/0/0, overriding the plain text authentication specified for area 1. <ul style="list-style-type: none"> • By default, all of the interfaces configured in the same area inherit the same authentication parameter values of the area.
Step 16	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Control Frequency That Same LSA Is Originated or Accepted for OSPF

This task explains how to tune the convergence time of OSPF routes in the routing table when many LSAs need to be flooded in a very short time interval.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. Perform Step 5 or Step 6 or both to control the frequency that the same LSA is originated or accepted.
5. **timers lsa refresh** *seconds*
6. **timers lsa min-arrival** *seconds*
7. **timers lsa group-pacing** *seconds*
8. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 Do one of the following:

- **router ospf** *process-name*
- **router ospfv3** *process-name*

Example:

```
RP/0/:router(config)# router ospf 1
```

or

```
RP/0/(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }

Example:

```
RP/0/(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IP address as the router ID.

Step 4 Perform Step 5 or Step 6 or both to control the frequency that the same LSA is originated or accepted.

Step 5 **timers lsa refresh** *seconds*

Example:

```
RP/0/(config-ospf)# timers lsa refresh 1800
```

Sets how often self-originated LSAs should be refreshed, in seconds.

- The default is 1800 seconds for both OSPF and OSPFv3.

Step 6 **timers lsa min-arrival** *seconds*

Example:


```
RP/0/(config-ospf)# timers lsa min-arrival 2
```

Limits the frequency that new processes of any particular OSPF Version 2 LSA can be accepted during flooding.

- The default is 1 second.

Step 7 **timers lsa group-pacing** *seconds*

Example:

```
RP/0/  
/CPU0:router(config-ospf)# timers lsa group-pacing 1000
```

Changes the interval at which OSPF link-state LSAs are collected into a group for flooding.

- The default is 240 seconds.

Step 8 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Creating a Virtual Link with MD5 Authentication to Area 0 for OSPF

This task explains how to create a virtual link to your backbone (area 0) and apply MD5 authentication. You must perform the steps described on both ABRs, one at each end of the virtual link. .



Note After you explicitly configure area parameter values, they are inherited by all interfaces bound to that area—unless you override the values and configure them explicitly for the interface. .

Before you begin

The following prerequisites must be met before creating a virtual link with MD5 authentication to area 0:

- You must have the router ID of the neighbor router at the opposite end of the link to configure the local router. You can execute the **show ospf** or **show ospfv3** command on the remote router to get its router ID.
- For a virtual link to be successful, you need a stable router ID at each end of the virtual link. You do not want them to be subject to change, which could happen if they are assigned by default. (See [OSPF Process and Router ID, on page 85](#) for an explanation of how the router ID is determined.) Therefore, we recommend that you perform one of the following tasks before configuring a virtual link:
 - Use the **router-id** command to set the router ID. This strategy is preferable.

- Configure a loopback interface so that the router has a stable router ID.
- Before configuring your virtual link for OSPF Version 2, you must decide whether to configure plain text authentication, MD5 authentication, or no authentication (which is the default). Your decision determines whether you need to perform additional tasks related to authentication.

SUMMARY STEPS

1. Do one of the following:
 - **show ospf** [*process-name*]
 - **show ospfv3** [*process-name*]
2. **configure**
3. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
4. **router-id** { *router-id* }
5. **area** *area-id*
6. **virtual-link** *router-id*
7. **authentication message-digest**
8. **message-digest-key** *key-id* **md5** { *key* | **clear** *key* | **encrypted** *key* }
9. Repeat all of the steps in this task on the ABR that is at the other end of the virtual link. Specify the same key ID and key that you specified for the virtual link on this router.
10. Use the **commit** or **end** command.
11. Do one of the following:
 - **show ospf** [*process-name*] [*area-id*] **virtual-links**
 - **show ospfv3** [*process-name*] **virtual-links**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Do one of the following: <ul style="list-style-type: none"> • show ospf [<i>process-name</i>] • show ospfv3 [<i>process-name</i>] Example: <pre>Router# show ospf</pre> or <pre>Router# show ospfv3</pre>	(Optional) Displays general information about OSPF routing processes. <ul style="list-style-type: none"> • The output displays the router ID of the local router. You need this router ID to configure the other end of the link.
Step 2	configure Example: <pre>RP/0/# configure</pre>	Enters mode.

	Command or Action	Purpose
Step 3	<p>Do one of the following:</p> <ul style="list-style-type: none"> • router ospf <i>process-name</i> • router ospfv3 <i>process-name</i> <p>Example:</p> <pre>Router(config)# router ospf 1</pre> <p>or</p> <pre>Router(config)# router ospfv3 1</pre>	<p>Enables OSPF routing for the specified routing process and places the router in router configuration mode.</p> <p>or</p> <p>Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.</p> <p>Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.</p>
Step 4	<p>router-id { <i>router-id</i> }</p> <p>Example:</p> <pre>Router(config-ospf)# router-id 192.168.4.3</pre>	<p>Configures a router ID for the OSPF process.</p> <p>Note We recommend using a stable IPv4 address as the router ID.</p>
Step 5	<p>area <i>area-id</i></p> <p>Example:</p> <pre>Router(config-ospf)# area 1</pre>	<p>Enters area configuration mode and configures a nonbackbone area for the OSPF process.</p> <ul style="list-style-type: none"> • The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.
Step 6	<p>virtual-link <i>router-id</i></p> <p>Example:</p> <pre>Router(config-ospf-ar)# virtual-link 10.3.4.5</pre>	<p>Defines an OSPF virtual link.</p> <ul style="list-style-type: none"> • See .
Step 7	<p>authentication message-digest</p> <p>Example:</p> <pre>Router(config-ospf-ar-vl)#authentication message-digest</pre>	<p>Selects MD5 authentication for this virtual link.</p>
Step 8	<p>message-digest-key <i>key-id</i> md5 { <i>key</i> clear <i>key</i> encrypted <i>key</i> }</p> <p>Example:</p> <pre>Router(config-ospf-ar-vl)#message-digest-key 4 md5 yourkey</pre>	<p>Defines an OSPF virtual link.</p> <ul style="list-style-type: none"> • See to understand a virtual link. • The <i>key-id</i> argument is a number in the range from 1 to 255. The <i>key</i> argument is an alphanumeric string of up to 16 characters. The routers at both ends of the virtual link must have the same key identifier and key to be able to route OSPF traffic. • The authentication-key <i>key</i> command is not supported for OSPFv3. • Once the key is encrypted it must remain encrypted.

	Command or Action	Purpose
Step 9	Repeat all of the steps in this task on the ABR that is at the other end of the virtual link. Specify the same key ID and key that you specified for the virtual link on this router.	—
Step 10	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.
Step 11	<p>Do one of the following:</p> <ul style="list-style-type: none"> • show ospf [<i>process-name</i>] [<i>area-id</i>] virtual-links • show ospfv3 [<i>process-name</i>] virtual-links <p>Example:</p> <pre>Router# show ospf 1 2 virtual-links</pre> <p>or</p> <pre>Router# show ospfv3 1 virtual-links</pre>	(Optional) Displays the parameters and the current state of OSPF virtual links.

Redistribute Routes into OSPF

This task redistributes routes from an IGP (could be a different OSPF process) into OSPF.

Before you begin

For information about configuring routing policy, see *Implementing Routing Policy on* module in the .

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **redistribute** *protocol* [*process-id*] { **level-1** | **level-1-2** | **level-2** } [**metric** *metric-value*] [**metric-type** *type-value*] [**match** { **external** [**1** | **2**] }] [**tag** *tag-value*] [**route-policy** *policy-name*]
5. Do one of the following:
 - **summary-prefix** *address mask* [**not-advertise**] [**tag** *tag*]

- **summary-prefix** *ipv6-prefix / prefix-length* [**not-advertise**] [**tag** *tag*]

6. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	Do one of the following: <ul style="list-style-type: none"> • router ospf <i>process-name</i> • router ospfv3 <i>process-name</i> Example: RP/0/(config)# router ospf 1 or RP/0/(config)# router ospfv3 1	Enables OSPF routing for the specified routing process and places the router in router configuration mode. or Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode. Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.
Step 3	router-id { <i>router-id</i> } Example: RRP/0/(config-ospf)# router-id 192.168.4.3	Configures a router ID for the OSPF process. Note We recommend using a stable IPv4 address as the router ID.
Step 4	redistribute <i>protocol</i> [<i>process-id</i>] { level-1 level-1-2 level-2 } [metric <i>metric-value</i>] [metric-type <i>type-value</i>] [match { external [1 2] } [tag <i>tag-value</i>] [route-policy <i>policy-name</i>] Example: RP/0/(config-ospf)# redistribute bgp 100 or RP/0/(config-router)# redistribute bgp 110	Redistributes OSPF routes from one routing domain to another routing domain. or Redistributes OSPFv3 routes from one routing domain to another routing domain. <ul style="list-style-type: none"> • This command causes the router to become an ASBR by definition. • OSPF tags all routes learned through redistribution as external. • The protocol and its process ID, if it has one, indicate the protocol being redistributed into OSPF. • The metric is the cost you assign to the external route. The default is 20 for all protocols except BGP, whose default metric is 1. • The OSPF example redistributes BGP autonomous system 1, Level 1 routes into OSPF as Type 2 external routes.

	Command or Action	Purpose
		<ul style="list-style-type: none"> The OSPFv3 example redistributes BGP autonomous system 1, Level 1 and 2 routes into OSPF. The external link type associated with the default route advertised into the OSPFv3 routing domain is the Type 1 external route. <p>Note RPL is not supported for OSPFv3.</p>
Step 5	<p>Do one of the following:</p> <ul style="list-style-type: none"> summary-prefix <i>address mask</i> [not-advertise] [<i>tag tag</i>] summary-prefix <i>ipv6-prefix / prefix-length</i> [not-advertise] [<i>tag tag</i>] <p>Example:</p> <pre>RP/0/(config-ospf)# summary-prefix 10.1.0.0 255.255.0.0</pre> <p>or</p> <pre>RP/0/(config-router)# summary-prefix 2010:11:22::/32</pre>	<p>(Optional) Creates aggregate addresses for OSPF.</p> <p>or</p> <p>(Optional) Creates aggregate addresses for OSPFv3.</p> <ul style="list-style-type: none"> This command provides external route summarization of the non-OSPF routes. External ranges that are being summarized should be contiguous. Summarization of overlapping ranges from two different routers could cause packets to be sent to the wrong destination. This command is optional. If you do not specify it, each route is included in the link-state database and advertised in LSAs. In the OSPFv2 example, the summary address 10.1.0.0 includes address 10.1.1.0, 10.1.2.0, 10.1.3.0, and so on. Only the address 10.1.0.0 is advertised in an external LSA. In the OSPFv3 example, the summary address 2010:11:22::/32 has addresses such as 2010:11:22:0:1000::1, 2010:11:22:0:2000:679:1, and so on. Only the address 2010:11:22::/32 is advertised in the external LSA.
Step 6	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> Yes — Saves configuration changes and exits the configuration session. No —Exits the configuration session without committing the configuration changes. Cancel —Remains in the configuration session, without committing the configuration changes.

Configuring OSPF Shortest Path First Throttling

This task explains how to configure SPF scheduling in millisecond intervals and potentially delay SPF calculations during times of network instability. This task is optional.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **timers throttle spf** *spf-start spf-hold spf-max-wait*
5. **area** *area-id*
6. **interface** *type interface-path-id*
7. Use the **commit** or **end** command.
8. Do one of the following:
 - **show ospf** [*process-name*]
 - **show ospfv3** [*process-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# <code>configure</code>	Enters mode.
Step 2	Do one of the following: <ul style="list-style-type: none"> • router ospf <i>process-name</i> • router ospfv3 <i>process-name</i> Example: RP/0/(config)# <code>router ospf 1</code> or RP/0/(config)# <code>router ospfv3 1</code>	Enables OSPF routing for the specified routing process and places the router in router configuration mode. or Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode. Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.
Step 3	router-id { <i>router-id</i> } Example: RP/0/(config-ospf)# <code>router-id 192.168.4.3</code>	Configures a router ID for the OSPF process. Note We recommend using a stable IPv4 address as the router ID.
Step 4	timers throttle spf <i>spf-start spf-hold spf-max-wait</i> Example:	Sets SPF throttling timers.

	Command or Action	Purpose
	RP/0/(config-ospf)# timers throttle spf 10 4800 90000	
Step 5	<p>area <i>area-id</i></p> <p>Example:</p> <pre>RP/0/(config-ospf)# area 0</pre>	<p>Enters area configuration mode and configures a backbone area.</p> <ul style="list-style-type: none"> The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.
Step 6	<p>interface <i>type interface-path-id</i></p> <p>Example:</p> <pre>RP/0/(config-ospf-ar)# interface GigabitEthernet 0/1/0/3</pre>	<p>Enters interface configuration mode and associates one or more interfaces to the area.</p>
Step 7	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> Yes — Saves configuration changes and exits the configuration session. No —Exits the configuration session without committing the configuration changes. Cancel —Remains in the configuration session, without committing the configuration changes.
Step 8	<p>Do one of the following:</p> <ul style="list-style-type: none"> show ospf [<i>process-name</i>] show ospfv3 [<i>process-name</i>] <p>Example:</p> <pre>RP/0/# show ospf 1</pre> <p>or</p> <pre>RP/0/# RP/0/RP0/CPU0:router# show ospfv3 2</pre>	<p>(Optional) Displays SPF throttling timers.</p>

Examples

In the following example, the **show ospf** command is used to verify that the initial SPF schedule delay time, minimum hold time, and maximum wait time are configured correctly. Additional details are displayed about the OSPF process, such as the router type and redistribution of routes.

```
show ospf 1
```



```

Routing Process "ospf 1" with ID 192.168.4.3
  Supports only single TOS(TOS0) routes
  Supports opaque LSA
  It is an autonomous system boundary router
  Redistributing External Routes from,
    ospf 2
  Initial SPF schedule delay 5 msec
  Minimum hold time between two consecutive SPF's 100 msec
  Maximum wait time between two consecutive SPF's 1000 msec
  Minimum LSA interval 5 secs. Minimum LSA arrival 1 sec
  Number of external LSA 0. Checksum Sum 00000000
  Number of opaque AS LSA 0. Checksum Sum 00000000
  Number of DCbitless external and opaque AS LSA 0
  Number of DoNotAge external and opaque AS LSA 0
  Number of areas in this router is 1. 1 normal 0 stub 0 nssa
  External flood list length 0
  Non-Stop Forwarding enabled

```



Note For a description of each output display field, see the **show ospf** command in the *OSPF Commands on* module in .

Configuring Nonstop Forwarding Specific to Cisco for OSPF Version 2

This task explains how to configure OSPF NSF specific to Cisco on your NSF-capable router. This task is optional.

Before you begin

OSPF NSF requires that all neighbor networking devices be NSF aware, which happens automatically after you install the Cisco IOS XR software image on the router. If an NSF-capable router discovers that it has non-NSF-aware neighbors on a particular network segment, it disables NSF capabilities for that segment. Other network segments composed entirely of NSF-capable or NSF-aware routers continue to provide NSF capabilities.



Note The following are restrictions when configuring nonstop forwarding:

- OSPF Cisco NSF for virtual links is not supported.
 - Neighbors must be NSF aware.
-

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **router-id** { *router-id* }
4. Do one of the following:
 - **nsf cisco**
 - **nsf cisco enforce global**

5. **nsf interval** *seconds*
6. **nsfflush-delay-time***seconds*
7. **nsflifetime***seconds*
8. **nsfietf**
9. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	router ospf <i>process-name</i> Example: RP/0/(config)# router ospf 1	Enables OSPF routing for the specified routing process and places the router in router configuration mode. Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.
Step 3	router-id { <i>router-id</i> } Example: RP/0/(config-ospf)# router-id 192.168.4.3	Configures a router ID for the OSPF process. Note We recommend using a stable IPv4 address as the router ID.
Step 4	Do one of the following: <ul style="list-style-type: none"> • nsf cisco • nsf cisco enforce global Example: RP/0/(config-ospf)# nsf cisco enforce global	Enables Cisco NSF operations for the OSPF process. <ul style="list-style-type: none"> • Use the nsf cisco command without the optional enforce and global keywords to abort the NSF restart mechanism on the interfaces of detected non-NSF neighbors and allow NSF neighbors to function properly. • Use the nsf cisco command with the optional enforce and global keywords if the router is expected to perform NSF during restart. However, if non-NSF neighbors are detected, NSF restart is canceled for the entire OSPF process.
Step 5	nsf interval <i>seconds</i> Example: RP/0/(config-ospf)# nsf interval 120	Sets the minimum time between NSF restart attempts. Note When you use this command, the OSPF process must be up for at least 90 seconds before OSPF attempts to perform an NSF restart.
Step 6	nsfflush-delay-time <i>seconds</i> Example:	Sets the maximum time allowed for external route learning in seconds.

	Command or Action	Purpose
	<code>RP/0/(config-ospf)#nsf flush-delay-time 1000</code>	
Step 7	<p>nsflifetime<i>seconds</i></p> <p>Example:</p> <pre>RP/0/(config-ospf)#nsf lifetime 90</pre>	Sets the maximum route lifetime of NSF following a restart in seconds.
Step 8	<p>nsfietf</p> <p>Example:</p> <pre>RP/0/(config-ospf)#nsf ietf</pre>	Enables ietf graceful restart.
Step 9	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Configuring OSPF Version 2 for MPLS Traffic Engineering

This task explains how to configure OSPF for MPLS TE. This task is optional.

For a description of the MPLS TE tasks and commands that allow you to configure the router to support tunnels, configure an MPLS tunnel that OSPF can use, and troubleshoot MPLS TE, see *Implementing MPLS Traffic Engineering* on module of the

Before you begin

Your network must support the following features before you enable MPLS TE for OSPF on your router:

- MPLS
- IP Cisco Express Forwarding (CEF)



Note You must enter the commands in the following task on every OSPF router in the traffic-engineered portion of your network.

SUMMARY STEPS

1. **configure**

2. **router ospf** *process-name*
3. **router-id** { *router-id* }
4. **mpls traffic-eng router-id** *interface-type interface-instance*
5. **area** *area-id*
6. **mpls traffic-eng**
7. **interface** *type interface-path-id*
8. Use the **commit** or **end** command.
9. **show ospf** [*process-name*] [*area-id*] **mpls traffic-eng** { **link** | **fragment** }

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	router ospf <i>process-name</i> Example: RP/0/(config)# router ospf 1	Enables OSPF routing for the specified routing process and places the router in router configuration mode. Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.
Step 3	router-id { <i>router-id</i> } Example: RP/0/(config-ospf)# router-id 192.168.4.3	Configures a router ID for the OSPF process. Note We recommend using a stable IPv4 address as the router ID.
Step 4	mpls traffic-eng router-id <i>interface-type interface-instance</i> Example: RP/0/(config-ospf)# mpls traffic-eng router-id loopback 0	(Optional) Specifies that the traffic engineering router identifier for the node is the IP address associated with a given interface. <ul style="list-style-type: none"> • This IP address is flooded to all nodes in TE LSAs. • For all traffic engineering tunnels originating at other nodes and ending at this node, you must set the tunnel destination to the traffic engineering router identifier of the destination node because that is the address that the traffic engineering topology database at the tunnel head uses for its path calculation. • We recommend that loopback interfaces be used for MPLS TE router ID because they are more stable than physical interfaces.
Step 5	area <i>area-id</i> Example: RP/0/(config-ospf)# area 0	Enters area configuration mode and configures an area for the OSPF process. <ul style="list-style-type: none"> • The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area

	Command or Action	Purpose
		1000 or area 0.0.3.232. However, you must choose one form or the other for an area.
Step 6	mpls traffic-eng Example: RP/0/(config-ospf)# mpls traffic-eng	Configures the MPLS TE under the OSPF area.
Step 7	interface <i>type interface-path-id</i> Example: RP/0/(config-ospf-ar)# interface interface loopback0	Enters interface configuration mode and associates one or more interfaces to the area.
Step 8	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.
Step 9	show ospf [<i>process-name</i>] [<i>area-id</i>] mpls traffic-eng { link fragment } Example: RP/0/# show ospf 1 0 mpls traffic-eng link	(Optional) Displays information about the links and fragments available on the local router for MPLS TE.

Configuring OSPFv3 Graceful Restart

This task explains how to configure a graceful restart for an OSPFv3 process. This task is optional.

SUMMARY STEPS

1. **configure**
2. **router ospfv3** *process-name*
3. **graceful-restart**
4. **graceful-restart lifetime**
5. **graceful-restart interval** *seconds*
6. **graceful-restart helper disable**
7. Use the **commit** or **end** command.
8. **show ospfv3** [*process-name* [*area-id*]] **database grace**

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	router ospfv3 <i>process-name</i> Example: RP/0/(config)# router ospfv3 test	Enters router configuration mode for OSPFv3. The process name is a WORD that uniquely identifies an OSPF routing process. The process name is any alphanumeric string no longer than 40 characters without spaces.
Step 3	graceful-restart Example: RP/0/(config-ospfv3)#graceful-restart	Enables graceful restart on the current router.
Step 4	graceful-restart lifetime Example: RP/0/(config-ospfv3)# graceful-restart lifetime 120	Specifies a maximum duration for a graceful restart. <ul style="list-style-type: none"> • The default lifetime is 95 seconds. • The range is 90 to 3600 seconds.
Step 5	graceful-restart interval <i>seconds</i> Example: RP/0/(config-ospfv3)# graceful-restart interval 120	Specifies the interval (minimal time) between graceful restarts on the current router. <ul style="list-style-type: none"> • The default value for the interval is 90 seconds. • The range is 90 to 3600 seconds.
Step 6	graceful-restart helper disable Example: RP/0/(config-ospfv3)# graceful-restart helper disable	Disables the helper capability.
Step 7	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

	Command or Action	Purpose
Step 8	show ospfv3 [<i>process-name</i> [<i>area-id</i>]] database grace Example: RP/0/# show ospfv3 1 database grace	Displays the state of the graceful restart link.

Configuring an OSPFv2 Sham Link

This task explains how to configure a provider edge (PE) router to establish an OSPFv2 sham link connection across a VPN backbone. This task is optional.

Before you begin

Before configuring a sham link in a Multiprotocol Label Switching (MPLS) VPN between provider edge (PE) routers, OSPF must be enabled as follows:

- Create an OSPF routing process.
- Configure a loopback interface that belongs to VRF and assign a IPv4 address with the host mask to it.
- Configure the sham link under the area submode.

See [Enable OSPF, on page 116](#) for information on these OSPF configuration prerequisites.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **vrf** *vrf-name*
4. **ipv4 address** *ip-address mask*
5. **end**
6. **router ospf** *instance-id*
7. **vrf** *vrf-name*
8. **router-id** { *router-id* }
9. **redistribute bgp** *process-id*
10. **area** *area-id*
11. **sham-link** *source-address destination-address*
12. **cost** *cost*
13. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.

	Command or Action	Purpose
Step 2	interface <i>type interface-path-id</i> Example: RP/0/(config)# interface loopback 3	Enters interface configuration mode.
Step 3	vrf <i>vrf-name</i> Example: RP/0/(config-if)# vrf vrf1	Assigns an interface to the VPN routing and forwarding (VRF) instance.
Step 4	ipv4 address <i>ip-address mask</i> Example: RP/0/(config-if)# ipv4 address 172.18.189.38 255.255.255.225	Assigns an IP address and subnet mask to the interface.
Step 5	end Example: RP/0/(config-if)# end	Saves configuration changes. When you issue the end command, the system prompts you to commit changes: <pre>Uncommitted changes found, commit them before exiting (yes/no/cancel)?[cancel]:</pre> <ul style="list-style-type: none"> • Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to <code>mode</code>. • Entering no exits the configuration session and returns the router to <code>mode</code> without committing the configuration changes. • Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes.
Step 6	router ospf <i>instance-id</i> Example: RP/0/(config)# router ospf isp	Enables OSPF routing for the specified routing process, and places the router in router configuration mode. In this example, the OSPF instance is called <code>isp</code> .
Step 7	vrf <i>vrf-name</i> Example: RP/0/(config-ospf)# vrf vrf1	Creates a VRF instance and enters VRF configuration mode.
Step 8	router-id { <i>router-id</i> } Example:	Configures a router ID for the OSPF process.

	Command or Action	Purpose
	RP/0/(config-ospf-vrf)# router-id 192.168.4.3	Note We recommend using a stable IPv4 address as the router ID.
Step 9	redistribute bgp <i>process-id</i> Example: RP/0/(config-ospf-vrf)# redistribute bgp 1	Redistributes OSPF routes from the one routing domain to another routing domain. <ul style="list-style-type: none"> • This command causes the router to become an ASBR by definition. • OSPF tags all routes learned through redistribution as external. • The protocol and its process ID, if it has one, indicate the protocol being redistributed into OSPF. • The BGP MED value is copied to the LSA metric field when BGP VPN routes are redistributed to OSPF.
Step 10	area <i>area-id</i> Example: RP/0/(config-ospf-vrf)# area 0	Enters area configuration mode and configures an area for the OSPF process. <ul style="list-style-type: none"> • The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area.
Step 11	sham-link <i>source-address destination-address</i> Example: RP/0/(config-ospf-vrf-ar)# sham-link 10.0.0.1 10.0.0.3	Configures a point-to-point unnumbered interface between two VPN sites.
Step 12	cost <i>cost</i> Example: RP/0/(config-ospf-vrf-ar-sl)# cost 76	Explicitly specifies the cost of sending a packet on an OSPF interface. The specified cost overrides the auto-costing calculated default value for interfaces.
Step 13	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Configure OSPFv2 OSPF SPF Prefix Prioritization

Perform this task to configure OSPFv2 OSPF SPF (shortest path first) prefix prioritization.

SUMMARY STEPS

1. **configure**
2. **prefix-set** *prefix-set name*
3. **route-policy** *route-policy name* **if destination in** *prefix-set name* **then set** **spf-priority** {critical | high | medium} **endif**
4. Use one of these commands:
 - **router ospf** *ospf-name*
 - **router ospfv3** *ospfv3-name*
5. **router ospf** *ospf name*
6. **spf prefix-priority route-policy** *route-policy name*
7. Use the **commit** or **end** command.
8. **show rpl route-policy** *route-policy name* **detail**

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 **prefix-set** *prefix-set name*

Example:

```
RP/0/(config)#prefix-set ospf-critical-prefixes
RP/0/(config-px)#66.0.0.0/16
RP/0/(config-px)#end-set
```

Configures the prefix set.

Step 3 **route-policy** *route-policy name* **if destination in** *prefix-set name* **then set** **spf-priority** {critical | high | medium} **endif**

Example:

```
RP/0/#route-policy ospf-spf-priority
RP/0/(config-rpl)#if destination in ospf-critical-prefixes then
  set spf-priority critical
endif
RP/0/(config-rpl)#end-policy
```

Configures route policy and sets OSPF SPF priority.

Step 4 Use one of these commands:

- **router ospf** *ospf-name*

- **router ospfv3** *ospfv3-name*

Example:

```
RP/0/# router ospf 1
```

Or

```
RP/0/# router ospfv3 1
```

Enters Router OSPF configuration mode.

Step 5 **router ospf** *ospf name***Example:**

```
RP/0/# router ospf 1
```

Enters Router OSPF configuration mode.

Step 6 **spf prefix-priority route-policy** *route-policy name***Example:**

```
RP/0/(config-ospf)# spf prefix-priority route-policy ospf-spf-priority
```

Or

```
RP/0/(config-ospfv3)#spf prefix-priority route-policy ospfv3-spf-priority
```

Configures SPF prefix-priority for the defined route policy.

Note Configure the **spf prefix-priority** command under router OSPF.

Step 7 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 8 **show rpl route-policy** *route-policy name* **detail****Example:**

```
RP/0/#show rpl route-policy ospf-spf-priority detail
prefix-set ospf-critical-prefixes
 66.0.0.0/16
end-set
!
route-policy ospf-spf-priority
  if destination in ospf-critical-prefixes then
    set spf-priority critical
  endif
end-policy
!
```

Displays the set SPF prefix priority.

Enabling Multicast-intact for OSPFv2

This optional task describes how to enable multicast-intact for OSPFv2 routes that use IPv4 addresses.

SUMMARY STEPS

1. **configure**
2. **router ospf** *instance-id*
3. **mpls traffic-eng multicast-intact**
4. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	router ospf <i>instance-id</i> Example: RP/0/(config)# router ospf isp	Enables OSPF routing for the specified routing process, and places the router in router configuration mode. In this example, the OSPF instance is called isp.
Step 3	mpls traffic-eng multicast-intact Example: RP/0/(config-ospf)# mpls traffic-eng multicast-intact	Enables multicast-intact.
Step 4	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Associating Interfaces to a VRF

This task explains how to associate an interface with a VPN Routing and Forwarding (VRF) instance.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **vrf** *vrf-name*
4. **area** *area-id*
5. **interface** *type interface-path-id*
6. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	router ospf <i>process-name</i> Example: RP/0/(config)# router ospf 1	Enables OSPF routing for the specified routing process and places the router in router configuration mode. Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.
Step 3	vrf <i>vrf-name</i> Example: RP/0/(config-ospf)# vrf vrf1	Creates a VRF instance and enters VRF configuration mode.
Step 4	area <i>area-id</i> Example: RP/0/(config-ospf-vrf)# area 0	Enters area configuration mode and configures an area for the OSPF process. • The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area.
Step 5	interface <i>type interface-path-id</i> Example: RP/0/(config-ospf-vrf-ar)# interface GigabitEthernet 0/0/0/0	Enters interface configuration mode and associates one or more interfaces to the VRF.
Step 6	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session.

	Command or Action	Purpose
		<p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Configuring OSPF as a Provider Edge to Customer Edge (PE-CE) Protocol

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **vrf** *vrf-name*
4. **router-id** { *router-id* }
5. **redistribute** *protocol* [*process-id*] { **level-1** | **level-1-2** | **level-2** } [**metric** *metric-value*] [**metric-type** *type-value*] [**match** { **external** [**1** | **2**] }] [**tag** *tag-value*] **route-policy** *policy-name*
6. **area** *area-id*
7. **interface** *type interface-path-id*
8. **exit**
9. **domain-id** [**secondary**] **type** { **0005** | **0105** | **0205** | **8005** } **value** *value*
10. **domain-tag** *tag*
11. **disable-dn-bit-check**
12. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# <code>configure</code>	Enters mode.
Step 2	router ospf <i>process-name</i> Example: RP/0/(config)# <code>router ospf 1</code>	Enables OSPF routing for the specified routing process and places the router in router configuration mode. Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.
Step 3	vrf <i>vrf-name</i> Example:	Creates a VRF instance and enters VRF configuration mode.

	Command or Action	Purpose
	RP/0/(config-ospf)# vrf vrf1	
Step 4	router-id { <i>router-id</i> } Example: RP/0/(config-ospf-vrf)# router-id 192.168.4.3	Configures a router ID for the OSPF process. Note We recommend using a stable IPv4 address as the router ID.
Step 5	redistribute <i>protocol</i> [<i>process-id</i>] { level-1 level-1-2 level-2 } [metric <i>metric-value</i>] [metric-type <i>type-value</i>] [match { external [1 2] }] [tag <i>tag-value</i>] [route-policy <i>policy-name</i>] Example: RP/0/(config-ospf-vrf)# redistribute bgp 1 level-1	Redistributes OSPF routes from one routing domain to another routing domain. <ul style="list-style-type: none"> • This command causes the router to become an ASBR by definition. • OSPF tags all routes learned through redistribution as external. • The protocol and its process ID, if it has one, indicate the protocol being redistributed into OSPF. • The metric is the cost you assign to the external route. The default is 20 for all protocols except BGP, whose default metric is 1. • The example shows the redistribution of BGP autonomous system 1, Level 1 routes into OSPF as Type 2 external routes.
Step 6	area <i>area-id</i> Example: RP/0/(config-ospf-vrf)# area 0	Enters area configuration mode and configures an area for the OSPF process. <ul style="list-style-type: none"> • The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area.
Step 7	interface <i>type interface-path-id</i> Example: RP/0/(config-ospf-vrf)# interface GigabitEthernet0/0/0/0	Enters interface configuration mode and associates one or more interfaces to the VRF.
Step 8	exit Example: RP/0/(config-if)# exit	Exits interface configuration mode.
Step 9	domain-id [secondary] type { 0005 / 0105 / 0205 / 8005 } value <i>value</i> Example:	Specifies the OSPF VRF domain ID. <ul style="list-style-type: none"> • The <i>value</i> argument is a six-octet hex number.

	Command or Action	Purpose
	RP/0/(config-ospf-vrf)# domain-id type 0105 value 1AF234	
Step 10	domain-tag <i>tag</i> Example: RP/0/(config-ospf-vrf)# domain-tag 234	Specifies the OSPF VRF domain tag. <ul style="list-style-type: none"> • The valid range for <i>tag</i> is 0 to 4294967295.
Step 11	disable-dn-bit-check Example: RP/0/(config-ospf-vrf)# disable-dn-bit-check	Specifies that down bits should be ignored.
Step 12	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Creating Multiple OSPF Instances (OSPF Process and a VRF)

This task explains how to create multiple OSPF instances. In this case, the instances are a normal OSPF instance and a VRF instance.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*
4. **interface** *type interface-path-id*
5. **exit**
6. **vrf** *vrf-name*
7. **area** *area-id*
8. **interface** *type interface-path-id*
9. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	router ospf process-name Example: RP/0/(config)# router ospf 1	Enables OSPF routing for the specified routing process and places the router in router configuration mode. Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.
Step 3	area area-id Example: RP/0/(config-ospf)# area 0	Enters area configuration mode and configures a backbone area. • The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.
Step 4	interface type interface-path-id Example: RP/0/(config-ospf-ar)# interface GigabitEthernet 0/1/0/3	Enters interface configuration mode and associates one or more interfaces to the area.
Step 5	exit Example: RP/0/(config-ospf-ar)# exit	Enters OSPF configuration mode.
Step 6	vrf vrf-name Example: RP/0/(config-ospf)# vrf vrf1	Creates a VRF instance and enters VRF configuration mode.
Step 7	area area-id Example: RP/0/(config-ospf-vrf)# area 0	Enters area configuration mode and configures an area for a VRF instance under the OSPF process. • The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area.
Step 8	interface type interface-path-id Example:	Enters interface configuration mode and associates one or more interfaces to the VRF.

	Command or Action	Purpose
	RP/0/ (config-ospf-vrf)# interface GigabitEthernet 0/0/0/0	
Step 9	Use the commit or end command.	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Configuring Multi-area Adjacency

This task explains how to create multiple areas on an OSPF primary interface.

Before you begin



Note You can configure multi-area adjacency on any interface where only two OSPF speakers are attached. In the case of native broadcast networks, the interface must be configured as an OSPF point-to-point type using the **network point-to-point** command to enable the interface for a multi-area adjacency.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*
4. **interface** *type interface-path-id*
5. **area** *area-id*
6. **multi-area-interface** *type interface-path-id*
7. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	<p>configure</p> <p>Example:</p> <p>RP/0/# configure</p>	Enters mode.

	Command or Action	Purpose
Step 2	<p>router ospf <i>process-name</i></p> <p>Example:</p> <pre>RP/0/(config)# router ospf 1</pre>	<p>Enables OSPF routing for the specified routing process and places the router in router configuration mode.</p> <p>Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.</p>
Step 3	<p>area <i>area-id</i></p> <p>Example:</p> <pre>RP/0/(config-ospf)# area 0</pre>	<p>Enters area configuration mode and configures a backbone area.</p> <ul style="list-style-type: none"> The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.
Step 4	<p>interface <i>type interface-path-id</i></p> <p>Example:</p> <pre>RP/0/(config-ospf-ar)# interface Serial 0/1/0/3</pre>	<p>Enters interface configuration mode and associates one or more interfaces to the area.</p>
Step 5	<p>area <i>area-id</i></p> <p>Example:</p> <pre>RP/0/(config-ospf)# area 1</pre>	<p>Enters area configuration mode and configures an area used for multiple area adjacency.</p> <ul style="list-style-type: none"> The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.
Step 6	<p>multi-area-interface <i>type interface-path-id</i></p> <p>Example:</p> <pre>RP/0/(config-ospf)# multi-area-interface Serial 0/1/0/3</pre>	<p>Enables multiple adjacencies for different OSPF areas and enters multi-area interface configuration mode</p>
Step 7	<p>Use the commit or end command.</p>	<p>commit —Saves the configuration changes and remains within the configuration session.</p> <p>end —Prompts user to take one of these actions:</p> <ul style="list-style-type: none"> Yes — Saves configuration changes and exits the configuration session. No —Exits the configuration session without committing the configuration changes. Cancel —Remains in the configuration session, without committing the configuration changes.

Configuring Label Distribution Protocol IGP Auto-configuration for OSPF

This task explains how to configure LDP auto-configuration for an OSPF instance.

Optionally, you can configure this feature for an area of an OSPF instance.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **mpls ldp auto-config**
4. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# <code>configure</code>	Enters mode.
Step 2	router ospf <i>process-name</i> Example: RP/0/(config)# <code>router ospf 1</code>	Enables OSPF routing for the specified routing process and places the router in router configuration mode. Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.
Step 3	mpls ldp auto-config Example: RP/0/(config-ospf)# <code>mpls ldp auto-config</code>	Enables LDP IGP interface auto-configuration for an OSPF instance. • Optionally, this command can be configured for an area of an OSPF instance.
Step 4	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Configuring LDP IGP Synchronization: OSPF

Perform this task to configure LDP IGP Synchronization under OSPF.



Note By default, there is no synchronization between LDP and IGPs.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. Use one of the following commands:
 - **mpls ldp sync**
 - **area** *area-id* **mpls ldp sync**
 - **area** *area-id* **interface** *name* **mpls ldp sync**
4. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# <code>configure</code>	Enters mode.
Step 2	router ospf <i>process-name</i> Example: RP/0/(config)# <code>router ospf 100</code>	Identifies the OSPF routing process and enters OSPF configuration mode.
Step 3	Use one of the following commands: <ul style="list-style-type: none"> • mpls ldp sync • area <i>area-id</i> mpls ldp sync • area <i>area-id</i> interface <i>name</i> mpls ldp sync Example: RP/0/(config-ospf)# <code>mpls ldp sync</code>	Enables LDP IGP synchronization on an interface.
Step 4	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Configuring Authentication Message Digest Management for OSPF

This task explains how to manage authentication of a keychain on the OSPF interface.

Before you begin

A valid keychain must be configured before this task can be attempted.

To learn how to configure a keychain and its associated attributes, see the *Implementing Key Chain Management on* module of the .

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **router-id** { *router-id* }
4. **area** *area-id*
5. **interface** *type interface-path-id*
6. **authentication** [**message-digest** *keychain* | **null**]
7. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	router ospf <i>process-name</i> Example: RP/0/(config)# router ospf 1	Enables OSPF routing for the specified routing process and places the router in router configuration mode. Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.
Step 3	router-id { <i>router-id</i> } Example: RP/0/(config-ospf)# router id 192.168.4.3	Configures a router ID for the OSPF process. Note We recommend using a stable IPv4 address as the router ID.
Step 4	area <i>area-id</i> Example: RP/0/(config-ospf)# area 1	Enters area configuration mode. The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.
Step 5	interface <i>type interface-path-id</i> Example:	Enters interface configuration mode and associates one or more interfaces to the area.

	Command or Action	Purpose
	RP/0/(config-ospf-ar)# interface GigabitEthernet0/4/0/1	
Step 6	authentication [message-digest keychain null] Example: RP/0/(config-ospf-ar-if)# authentication message-digest keychain ospf_int1	Configures an MD5 keychain. Note In the example, the <i>ospf_int1</i> keychain must be configured before you attempt this step.
Step 7	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Examples

The following example shows how to configure the keychain *ospf_intf_1* that contains five key IDs. Each key ID is configured with different **send-lifetime** values; however, all key IDs specify the same text string for the key.

```
key chain ospf_intf_1
key 1
send-lifetime 11:30:30 May 1 2007 duration 600
cryptographic-algorithm MD5T
key-string clear ospf_intf_1
key 2
send-lifetime 11:40:30 May 1 2007 duration 600
cryptographic-algorithm MD5
key-string clear ospf_intf_1
key 3
send-lifetime 11:50:30 May 1 2007 duration 600
cryptographic-algorithm MD5
key-string clear ospf_intf_1
key 4
send-lifetime 12:00:30 May 1 2007 duration 600
cryptographic-algorithm MD5
key-string clear ospf_intf_1
key 5
send-lifetime 12:10:30 May 1 2007 duration 600
cryptographic-algorithm MD5
key-string clear ospf_intf_1
```

The following example shows that keychain authentication is enabled on the Gigabit Ethernet 0/4/0/1 interface:

```
show ospf 1 interface GigabitEthernet0/4/0/1
```

```
GigabitEthernet0/4/0/1 is up, line protocol is up
Internet Address 100.10.10.2/24, Area 0
Process ID 1, Router ID 2.2.2.1, Network Type BROADCAST, Cost: 1
Transmit Delay is 1 sec, State DR, Priority 1
Designated Router (ID) 2.2.2.1, Interface address 100.10.10.2
Backup Designated router (ID) 1.1.1.1, Interface address 100.10.10.1
Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
  Hello due in 00:00:02
Index 3/3, flood queue length 0
Next 0(0)/0(0)
Last flood scan length is 2, maximum is 16
Last flood scan time is 0 msec, maximum is 0 msec
Neighbor Count is 1, Adjacent neighbor count is 1
  Adjacent with neighbor 1.1.1.1 (Backup Designated Router)
Suppress hello for 0 neighbor(s)
Keychain-based authentication enabled
  Key id used is 3
Multi-area interface Count is 0
```

The following example shows output for configured keys that are active:

```
show key chain ospf_intf_1

Key-chain: ospf_intf_1/ -

Key 1 -- text "0700325C4836100B0314345D"
  cryptographic-algorithm -- MD5
  Send lifetime: 11:30:30, 01 May 2007 - (Duration) 600
  Accept lifetime: Not configured
Key 2 -- text "10411A0903281B051802157A"
  cryptographic-algorithm -- MD5
  Send lifetime: 11:40:30, 01 May 2007 - (Duration) 600
  Accept lifetime: Not configured
Key 3 -- text "06091C314A71001711112D5A"
  cryptographic-algorithm -- MD5
  Send lifetime: 11:50:30, 01 May 2007 - (Duration) 600 [Valid now]
  Accept lifetime: Not configured
Key 4 -- text "151D181C0215222A3C350A73"
  cryptographic-algorithm -- MD5
  Send lifetime: 12:00:30, 01 May 2007 - (Duration) 600
  Accept lifetime: Not configured
Key 5 -- text "151D181C0215222A3C350A73"
  cryptographic-algorithm -- MD5
  Send lifetime: 12:10:30, 01 May 2007 - (Duration) 600
  Accept lifetime: Not configured
```

Configuring Generalized TTL Security Mechanism (GTSM) for OSPF

This task explains how to set the security time-to-live mechanism on an interface for GTSM.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **router-id** { *router-id* }
4. **log adjacency changes** [**detail** | **disable**]
5. **nsf** { **cisco** [**enforce global**] | **ietf** [**helper disable**]}

6. **timers throttle spf** *spf-start spf-hold spf-max-wait*
7. **area** *area-id*
8. **interface** *type interface-path-id*
9. **security ttl** [**disable** | **hops** *hop-count*]
10. Use the **commit** or **end** command.
11. **show ospf** [*process-name*] [*area-id*] **interface** [*type interface-path-id*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# <code>configure</code>	Enters mode.
Step 2	router ospf <i>process-name</i> Example: RP/0/(config)# <code>router ospf 1</code>	Enables OSPF routing for the specified routing process and places the router in router configuration mode. Note The <i>process-name</i> argument is any alphanumeric string no longer than 40 characters.
Step 3	router-id { <i>router-id</i> } Example: RP/0/(config-ospf)# <code>router id 10.10.10.100</code>	Configures a router ID for the OSPF process. Note We recommend using a stable IPv4 address as the router ID.
Step 4	log adjacency changes [detail disable] Example: RP/0/(config-ospf-ar-if)# <code>log adjacency changes detail</code>	(Optional) Requests notification of neighbor changes. <ul style="list-style-type: none"> • By default, this feature is enabled. • The messages generated by neighbor changes are considered notifications, which are categorized as severity Level 5 in the logging console command. The logging console command controls which severity level of messages are sent to the console. By default, all severity level messages are sent.
Step 5	nsf { cisco [enforce global] ietf [helper disable] } Example: RP/0/(config-ospf)# <code>nsf ietf</code>	(Optional) Configures NSF OSPF protocol. The example enables graceful restart.
Step 6	timers throttle spf <i>spf-start spf-hold spf-max-wait</i> Example: RP/0/(config-ospf)# <code>timers throttle spf 500 500 10000</code>	(Optional) Sets SPF throttling timers.
Step 7	area <i>area-id</i>	Enters area configuration mode.

	Command or Action	Purpose
	Example: <pre>RP/0/(config-ospf)# area 1</pre>	The <i>area-id</i> argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.
Step 8	interface <i>type interface-path-id</i> Example: <pre>RP/0/(config-ospf-ar)# interface GigabitEthernet0/5/0/0</pre>	Enters interface configuration mode and associates one or more interfaces to the area.
Step 9	security ttl [disable hops <i>hop-count</i>] Example: <pre>RP/0/(config-ospf-ar-if)# security ttl hops 2</pre>	Sets the security TTL value in the IP header for OSPF packets.
Step 10	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.
Step 11	show ospf [<i>process-name</i>] [<i>area-id</i>] interface [<i>type interface-path-id</i>] Example: <pre>RP/0/# show ospf 1 interface GigabitEthernet0/5/0/0</pre>	Displays OSPF interface information.

Examples

The following is sample output that displays the GTSM security TTL value configured on an OSPF interface:

```
show ospf 1 interface GigabitEthernet0/5/0/0
```

```
GigabitEthernet0/5/0/0 is up, line protocol is up
 Internet Address 120.10.10.1/24, Area 0
 Process ID 1, Router ID 100.100.100.100, Network Type BROADCAST, Cost: 1
 Transmit Delay is 1 sec, State BDR, Priority 1
 TTL security enabled, hop count 2
 Designated Router (ID) 102.102.102.102, Interface address 120.10.10.3
 Backup Designated router (ID) 100.100.100.100, Interface address 120.10.10.1
 Flush timer for old DR LSA due in 00:02:36
```

```

Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
  Hello due in 00:00:05
Index 1/1, flood queue length 0
Next 0(0)/0(0)
Last flood scan length is 1, maximum is 4
Last flood scan time is 0 msec, maximum is 0 msec
Neighbor Count is 1, Adjacent neighbor count is 1
  Adjacent with neighbor 102.102.102.102 (Designated Router)
Suppress hello for 0 neighbor(s)
Multi-area interface Count is 0

```

Verify OSPF Configuration and Operation

This task explains how to verify the configuration and operation of OSPF.

SUMMARY STEPS

1. **show** { **ospf** | **ospfv3** } [*process-name*]
2. **show** { **ospf** | **ospfv3** } [*process-name*] **border-routers** [*router-id*]
3. **show** { **ospf** | **ospfv3** } [*process-name*] **database**
4. **show** { **ospf** | **ospfv3** } [*process-name*] [*area-id*] **flood-list interface** *type interface-path-id*
5. **show** { **ospf** | **ospfv3** } [*process-name*] [**vrf** *vrf-name*] [*area-id*] **interface** [*type interface-path-id*]
6. **show** { **ospf** | **ospfv3** } [*process-name*] [*area-id*] **neighbor** [*type interface-path-id*] [*neighbor-id*] [**detail**]
7. **clear** { **ospf** | **ospfv3** } [*process-name*] **process**
8. **clear** { **ospf** | **ospfv3** } [*process-name*] **redistribution**
9. **clear** { **ospf** | **ospfv3** } [*process-name*] **routes**
10. **clear** { **ospf** | **ospfv3** } [*process-name*] **vrf** [*vrf-name* | **all**] { **process** | **redistribution** | **routes** | **statistics** [**interface** *type interface-path-id* | **message-queue** | **neighbor**] }
11. **clear** { **ospf** | **ospfv3** } [*process-name*] **statistics** [**neighbor** [*type interface-path-id*]] [*ip-address*]

DETAILED STEPS

Step 1 **show** { **ospf** | **ospfv3** } [*process-name*]

Example:

```
RP/0/# show ospf group1
```

(Optional) Displays general information about OSPF routing processes.

Step 2 **show** { **ospf** | **ospfv3** } [*process-name*] **border-routers** [*router-id*]

Example:

```
RP/0/# show ospf group1 border-routers
```

(Optional) Displays the internal OSPF routing table entries to an ABR and ASBR.

Step 3 **show** { **ospf** | **ospfv3** } [*process-name*] **database**

Example:

```
RP/0/# show ospf group2 database
```

(Optional) Displays the lists of information related to the OSPF database for a specific router.

- The various forms of this command deliver information about different OSPF LSAs.

Step 4 **show { ospf | ospfv3 } [process-name] [area-id] flood-list interface type interface-path-id**

Example:

```
RP/0/# show ospf 100 flood-list interface TenGigE 0/3/0/0
```

(Optional) Displays a list of OSPF LSAs waiting to be flooded over an interface.

Step 5 **show { ospf | ospfv3 } [process-name] [vrf vrf-name] [area-id] interface [type interface-path-id]**

Example:

```
RP/0/# show ospf 100 interface TenGigE 0/3/0/0
```

(Optional) Displays OSPF interface information.

Step 6 **show { ospf | ospfv3 } [process-name] [area-id] neighbor [type interface-path-id] [neighbor-id] [detail]**

Example:

```
RP/0/# show ospf 100 neighbor
```

(Optional) Displays OSPF neighbor information on an individual interface basis.

Step 7 **clear { ospf | ospfv3 } [process-name] process**

Example:

```
RP/0/  
/CPU0:router# clear ospf 100 process
```

(Optional) Resets an OSPF router process without stopping and restarting it.

Step 8 **clear { ospf | ospfv3 } [process-name] redistribution**

Example:

```
RP/0/#clear ospf 100 redistribution
```

Clears OSPF route redistribution.

Step 9 **clear { ospf | ospfv3 } [process-name] routes**

Example:

```
RP/0/#clear ospf 100 routes
```

Clears OSPF route table.

Step 10 **clear { ospf | ospfv3 } [process-name] vrf [vrf-name | all] { process | redistribution | routes | statistics } [interface type interface-path-id | message-queue | neighbor] }**

Example:

```
RP/0/#clear ospf 100 vrf vrf_1 process
```

Clears OSPF route table.

Step 11 **clear** { **ospf** | **ospfv3** } [*process-name*] **statistics** [**neighbor** [*type interface-path-id*] [*ip-address*]]

Example:

```
RP/0/# clear ospf 100 statistics
```

(Optional) Clears the OSPF statistics of neighbor state transitions.

Configuring IP Fast Reroute Loop-free Alternate

This task describes how to enable the IP fast reroute (IPFRR) per-link loop-free alternate (LFA) computation to converge traffic flows around link failures.

To enable protection on broadcast links, IPFRR and bidirectional forwarding detection (BFD) must be enabled on the interface under OSPF.

Enabling IPFRR LFA

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*
4. **interface** *type interface-path-id*
5. **fast-reroute per-link** { **enable** | **disable** }
6. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	router ospf <i>process-name</i> Example: RP/0/(config)# router ospf	Enables OSPF routing for the specified routing process and places the router in router configuration mode.
Step 3	area <i>area-id</i> Example:	Enters area configuration mode.

	Command or Action	Purpose
	RP/0/ (config-ospf) #area 1	
Step 4	interface <i>type interface-path-id</i> Example: RP/0/ (config-ospf-ar) # interface GigabitEthernet0/5/0/0	Enters interface configuration mode and associates one or more interfaces to the area. .
Step 5	fast-reroute per-link { enable disable } Example: RP/0/ (config-ospf-ar) #fast-reroute per-link enable	Enables or disables per-link LFA computation for the interface.
Step 6	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Excluding an Interface From IP Fast Reroute Per-link Computation

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*
4. **interface** *type interface-path-id*
5. **fast-reroute per-link exclude interface** *type interface-path-id*
6. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.

	Command or Action	Purpose
Step 2	router ospf <i>process-name</i> Example: RP/0/(config)# router ospf	Enables the OSPF routing for the specified routing process and places the router in router configuration mode.
Step 3	area <i>area-id</i> Example: RP/0/(config)#area area-id	Enters area configuration mode.
Step 4	interface <i>type interface-path-id</i> Example: RP/0/(config-ospf)#interface type interface-path-id	Enters interface configuration mode and associates one or more interfaces to the area.
Step 5	fast-reroute per-link exclude interface <i>type interface-path-id</i> Example: RP/0/(config-ospf-ar)# fast-reroute per-link exclude interface GigabitEthernet0/5/0/1	Excludes an interface from IP fast reroute per-link computation.
Step 6	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

Enabling OSPF Interaction with SRMS Server

To enable OSPF interaction with SRMS server:

SUMMARY STEPS

1. **configure**
2. **router ospf** *instance-id*
3. **segment-routing mpls**
4. **segment-routing forwarding mpls**
5. **segment-routing prefix-sid-mapadvertise-local**
6. **segment-routing sr-preferprefix-list**[*acl-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure Example: RP/0/# configure	Enters mode.
Step 2	router ospf <i>instance-id</i> Example: RP/0/(config)# router ospf isp	Enables OSPF routing for the specified routing instance, and places the router in router configuration mode.
Step 3	segment-routing mpls Example: RP/0/(config-ospf)# segment-routing mpls	
Step 4	segment-routing forwarding mpls Example: RP/0/(config-ospf)# segment-routing forwarding mpls	Enables SR forwarding on all interfaces where this instance OSPF is enabled.
Step 5	segment-routing prefix-sid-map advertise-local Example: RP/0/(config-ospf)# segment-routing prefix-sid-map advertise local	Enables server functionality and allows OSPF to advertise the local mapping entries using area-scope flooding. The flooding is limited to areas where segment-routing is enabled. Disabled by default.
Step 6	segment-routing sr-prefer prefix-list[<i>acl-name</i>] Example: RP/0/(config-ospf)# segment-routing sr-prefer prefix-list foo	Enables OSPF to communicate to the routing information base (RIB) that SR labels are preferred to LDP labels. If ACL is used, OSPF signals the preference of SR labels over LDP labels for prefixes that match ACL. If ACL is not used, OSPF signals the preference of SR labels for all prefixes.

Example

The following example shows how OSPF advertises local mapping entries using area-flooding scope.

```

ipv4 prefix-list foo
permit 2.2.2.2/32
!
router ospf 1
router-id 1.1.1.1
segment-routing mpls
segment-routing forwarding mpls
segment-routing prefix-sid-map receive
segment-routing prefix-sid-map advertise-local
segment-routing sr-prefer prefix-list foo
area 0
interface Loopback0
prefix-sid index 1
!

```



```

interface GigabitEthernet0/0/0/0
!
interface GigabitEthernet0/2/0/0
!
interface GigabitEthernet0/2/0/3
!
!
area 1
interface GigabitEthernet0/2/0/7
!

```

Configuration Examples for Implementing OSPF

This section provides the following configuration examples:

Cisco IOS XR Software for OSPF Version 2 Configuration: Example

The following example shows how an OSPF interface is configured for an area in Cisco IOS XR Software. area 0 must be explicitly configured with the **area** command and all interfaces that are in the range from 10.1.2.0 to 10.1.2.255 are bound to area 0. Interfaces are configured with the **interface** command (while the router is in area configuration mode) and the **area** keyword is not included in the interface statement.

Cisco IOS XR Software Configuration

```

interface GigabitEthernet 0/3/0/0
 ip address 10.1.2.1 255.255.255.255
 negotiation auto
!
router ospf 1
router-id 10.2.3.4
 area 0
   interface GigabitEthernet 0/3/0/0
!
!

```

The following example shows how OSPF interface parameters are configured for an area in Cisco IOS XR software.

In Cisco IOS XR software, OSPF interface-specific parameters are configured in interface configuration mode and explicitly defined for area 0. In addition, the **ip ospf** keywords are no longer required.

Cisco IOS XR Software Configuration

```

interface GigabitEthernet 0/3/0/0
 ip address 10.1.2.1 255.255.255.0
 negotiation auto
!
router ospf 1
router-id 10.2.3.4
area 0
interface GigabitEthernet 0/3/0/0
 cost 77
 mtu-ignore
 authentication message-digest

```

```

message-digest-key 1 md5 0 test
!
!

```

The following example shows the hierarchical CLI structure of Cisco IOS XR software:

In Cisco IOS XR software, OSPF areas must be explicitly configured, and interfaces configured under the area configuration mode are explicitly bound to that area. In this example, interface 10.1.2.0/24 is bound to area 0 and interface 10.1.3.0/24 is bound to area 1.

Cisco IOS XR Software Configuration

```

interface GigabitEthernet 0/3/0/0
 ip address 10.1.2.1 255.255.255.0
 negotiation auto
!
interface GigabitEthernet 0/3/0/1
 ip address 10.1.3.1 255.255.255.0
 negotiation auto
!
router ospf 1
 router-id 10.2.3.4
 area 0
  interface GigabitEthernet 0/3/0/0
  !
 area 1
  interface GigabitEthernet 0/3/0/1
  !
!
!

```

CLI Inheritance and Precedence for OSPF Version 2: Example

The following example configures the cost parameter at different hierarchical levels of the OSPF topology, and illustrates how the parameter is inherited and how only one setting takes precedence. According to the precedence rule, the most explicit configuration is used.

The cost parameter is set to 5 in router configuration mode for the OSPF process. Area 1 sets the cost to 15 and area 6 sets the cost to 30. All interfaces in area 0 inherit a cost of 5 from the OSPF process because the cost was not set in area 0 or its interfaces.

In area 1, every interface has a cost of 15 because the cost is set in area 1 and 15 overrides the value 5 that was set in router configuration mode.

Area 4 does not set the cost, but GigabitEthernet interface 01/0/2 sets the cost to 20. The remaining interfaces in area 4 have a cost of 5 that is inherited from the OSPF process.

Area 6 sets the cost to 30, which is inherited by GigabitEthernet interfaces 0/1/0/3 and 0/2/0/3. GigabitEthernet interface 0/3/0/3 uses the cost of 1, which is set in interface configuration mode.

```

router ospf 1
 router-id 10.5.4.3
 cost 5
 area 0
  interface GigabitEthernet 0/1/0/0
  !
  interface GigabitEthernet 0/2/0/0
  !
  interface GigabitEthernet 0/3/0/0
  !

```

```

!
area 1
cost 15
interface GigabitEthernet 0/1/0/1
!
interface GigabitEthernet 0/2/0/1
!
interface GigabitEthernet 0/3/0/1
!
!
area 4
interface GigabitEthernet 0/1/0/2
cost 20
!
interface GigabitEthernet 0/2/0/2
!
interface GigabitEthernet 0/3/0/2
!
!
area 6
cost 30
interface GigabitEthernet 0/1/0/3
!
interface GigabitEthernet 0/2/0/3
!
interface GigabitEthernet 0/3/0/3
cost 1
!
!

```

MPLS TE for OSPF Version 2: Example

The following example shows how to configure the OSPF portion of MPLS TE. However, you still need to build an MPLS TE topology and create an MPLS TE tunnel. See the for information.

In this example, loopback interface 0 is associated with area 0 and MPLS TE is configured within area 0.

```

interface Loopback 0
address 10.10.10.10 255.255.255.0
!
interface GigabitEthernet 0/2/0/0
address 10.1.2.2 255.255.255.0
!
router ospf 1
router-id 10.10.10.10
nsf
auto-cost reference-bandwidth 10000
mpls traffic-eng router-id Loopback 0
area 0
mpls traffic-eng
interface GigabitEthernet 0/2/0/0
interface Loopback 0

```

ABR with Summarization for OSPFv3: Example

The following example shows the prefix range 2300::/16 summarized from area 1 into the backbone:

```

router ospfv3 1

```

```

router-id 192.168.0.217
area 0
 interface GigabitEthernet 0/2/0/1
area 1
 range 2300::/16
 interface GigabitEthernet 0/2/0/0

```

ABR Stub Area for OSPFv3: Example

The following example shows that area 1 is configured as a stub area:

```

router ospfv3 1
router-id 10.0.0.217
area 0
 interface GigabitEthernet 0/2/0/1
area 1
 stub
 interface GigabitEthernet 0/2/0/0

```

ABR Totally Stub Area for OSPFv3: Example

The following example shows that area 1 is configured as a totally stub area:

```

router ospfv3 1
router-id 10.0.0.217
area 0
 interface GigabitEthernet 0/2/0/1
area 1
 stub no-summary
 interface GigabitEthernet 0/2/0/0

```

Configuring OSPFv2 SPF Prefix Prioritization: Example

This example shows how to configure /32 prefixes as medium-priority, in general, in addition to placing some /32 and /24 prefixes in critical-priority and high-priority queues:

```

prefix-set ospf-critical-prefixes
 192.41.5.41/32,
 11.1.3.0/24,
 192.168.0.44/32
end-set
!
prefix-set ospf-high-prefixes
 44.4.10.0/24,
 192.41.4.41/32,
 41.4.41.41/32
end-set
!
prefix-set ospf-medium-prefixes
 0.0.0.0/0 ge 32
end-set
!

route-policy ospf-priority
 if destination in ospf-high-prefixes then
  set spf-priority high

```

```

else
  if destination in ospf-critical-prefixes then
    set spf-priority critical
  else
    if destination in ospf-medium-prefixes then
      set spf-priority medium
    endif
  endif
endif
end-policy

```

OSPFv2

```

router ospf 1
  spf prefix-priority route-policy ospf-priority
  area 0
    interface GigabitEthernet0/3/0/0
    !
  area 3
    interface GigabitEthernet0/2/0/0
    !
  area 8
    interface GigabitEthernet0/2/0/0.590

```

Route Redistribution for OSPFv3: Example

The following example uses prefix lists to limit the routes redistributed from other protocols.

Only routes with 9898:1000 in the upper 32 bits and with prefix lengths from 32 to 64 are redistributed from BGP 42. Only routes *not* matching this pattern are redistributed from BGP 1956.

```

ipv6 prefix-list list1
  seq 10 permit 9898:1000::/32 ge 32 le 64
ipv6 prefix-list list2
  seq 10 deny 9898:1000::/32 ge 32 le 64
  seq 20 permit ::/0 le 128
router ospfv3 1
  router-id 10.0.0.217
  redistribute bgp 42
  redistribute bgp 1956
  distribute-list prefix-list list1 out bgp 42
  distribute-list prefix-list list2 out bgp 1956
  area 1
    interface GigabitEthernet 0/2/0/0

```

Virtual Link Configured Through Area 1 for OSPFv3: Example

This example shows how to set up a virtual link to connect the backbone through area 1 for the OSPFv3 topology that consists of areas 0 and 1 and virtual links 10.0.0.217 and 10.0.0.212:

ABR 1 Configuration

```

router ospfv3 1
  router-id 10.0.0.217

```

```

area 0
 interface GigabitEthernet 0/2/0/1
area 1
 virtual-link 10.0.0.212
 interface GigabitEthernet 0/2/0/0

```

ABR 2 Configuration

```

router ospfv3 1
 router-id 10.0.0.212
 area 0
 interface GigabitEthernet 0/3/0/1
 area 1
 virtual-link 10.0.0.217
 interface GigabitEthernet 0/2/0/0

```

Check the virtual links:

```

show ospfv3 virtual-links
Mon Dec 17 11:18:29.249 EST

```

```

Virtual Link OSPF_VL0 to router 192.168.0.4 is up
Interface ID 1000000, IPv6 address 13:13:13::4
Run as demand circuit
DoNotAge LSA allowed.
Transit area 1, via interface GigabitEthernet0/0/0/0, Cost of using 2
Transmit Delay is 1 sec, State POINT_TO_POINT,
Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
Hello due in 00:00:06
Adjacency State INIT (Hello suppressed)
Index 0/0/0, retransmission queue length 0, number of retransmission 0
First 0(0)/0(0)/0(0) Next 0(0)/0(0)/0(0)
Last retransmission scan length is 0, maximum is 0
Last retransmission scan time is 0 msec, maximum is 0 msec

```

Virtual Link Configured with MD5 Authentication for OSPF Version 2: Example

The following examples show how to configure a virtual link to your backbone and apply MD5 authentication. You must perform the steps described on both ABRs at each end of the virtual link.

After you explicitly configure the ABRs, the configuration is inherited by all interfaces bound to that area—unless you override the values and configure them explicitly for the interface.

To understand virtual links, see [Virtual Link and Transit Area for OSPF, on page 92](#).

In this example, all interfaces on router ABR1 use MD5 authentication:

```

router ospf ABR1
 router-id 10.10.10.10
 authentication message-digest
 message-digest-key 100 md5 0 cisco
 area 0
 interface GigabitEthernet 0/2/0/1
 interface GigabitEthernet 0/3/0/0
 area 1
 interface GigabitEthernet 0/3/0/1
 virtual-link 10.10.5.5
!
!

```

In this example, only area 1 interfaces on router ABR3 use MD5 authentication:

```

router ospf ABR2
router-id 10.10.5.5
area 0
area 1
  authentication message-digest
  message-digest-key 100 md5 0 cisco
  interface GigabitEthernet 0/9/0/1
  virtual-link 10.10.10.10
area 3
  interface Loopback 0
  interface GigabitEthernet 0/9/0/0
!
```

VPN Backbone and Sham Link Configured for OSPF Version 2: Example

The following examples show how to configure a provider edge (PE) router to establish a VPN backbone and sham link connection:

```

logging console debugging
vrf vrf_1
  address-family ipv4 unicast
  import route-target
    100:1
  !
  export route-target
    100:1
  !
!
!
interface Loopback0
  ipv4 address 2.2.2.1 255.255.255.255
  !
interface Loopback1
  vrf vrf_1
  ipv4 address 10.0.1.3 255.255.255.255
  !
interface GigabitEthernet0/2/0/2
  vrf vrf_1
  ipv4 address 100.10.10.2 255.255.255.0
  !
interface GigabitEthernet0/2/0/3
  ipv4 address 100.20.10.2 255.255.255.0
  !
!
route-policy pass-all
  pass
end-policy
!
router ospf 1
  log adjacency changes
  router-id 2.2.2.2
  vrf vrf_1
  router-id 22.22.22.2
  domain-id type 0005 value 111122223333
  domain-tag 140
  nsf ietf
  redistribute bgp 10
  area 0
  sham-link 10.0.1.3 10.0.0.101
  !
  interface GigabitEthernet0/2/0/2
```

```

!
!
!
!
router ospf 2
  router-id 2.22.2.22
  area 0
    interface Loopback0
    !
    interface GigabitEthernet0/2/0/3
    !
    !
!
router bgp 10
  bgp router-id 2.2.2.1
  bgp graceful-restart restart-time 300
  bgp graceful-restart
  address-family ipv4 unicast
    redistribute connected
  !
  address-family vpnv4 unicast
  !
  neighbor 2.2.2.2
  remote-as 10
  update-source Loopback0
  address-family ipv4 unicast
  !
  address-family vpnv4 unicast
  !
!
vrf vrf_1
  rd 100:1
  address-family ipv4 unicast
    redistribute connected route-policy pass-all
    redistribute ospf 1 match internal external
  !
!
!
mpls ldp
  router-id 2.2.2.1
  interface GigabitEthernet0/2/0/3
  !
!

```

The show command for the configuration is as follows:

```

show ospf vrf all-inclusive sham-links
Mon Dec 17 10:27:41.815 EST

```

```

Sham Links for OSPF 1, VRF vrf1

```

```

Sham Link OSPF_SL0 to address 3.3.3.3 is up
Area 1, source address 1.1.1.1
IfIndex = 3
Run as demand circuit
DoNotAge LSA allowed., Cost of using 2
Transmit Delay is 1 sec, State POINT_TO_POINT,
Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
Hello due in 00:00:08:911
Adjacency State FULL (Hello suppressed)
Number of DBD retrans during last exchange 0
Index 2/2, retransmission queue length 0, number of retransmission 1
First 0(0)/0(0) Next 0(0)/0(0)
Last retransmission scan length is 1, maximum is 1

```



```

Last retransmission scan time is 0 msec, maximum is 0 msec
Keychain-based authentication enabled
Keychain name key1
Key id used is 1
Cryptographic algorithm MD5_16

```

OSPF v3 Sham Links

The following example shows a configuration for ospfv3 sham links:

```

router ospfv3 1
vrf vrf1
auto-cost reference-bandwidth 1000
router-id 1.1.1.1
redistribute bgp 100 route-policy vrf1_rpl
area 1
sham-link 1111::1111 3333::3333
cost 2
!

```

The show command for the configuration is as follows:

```

show ospfv3 vrf all-inclusive sham-links
Mon Dec 17 11:06:05.192 EST

Sham Links for OSPFv3 1, VRF vrf1

Sham Link OSPF_SL0 to address 3333::3333 is up
Area 1, source address 1111::1111
IfIndex = 3
Run as demand circuit
DoNotAge LSA allowed., Cost of using 2
Transmit Delay is 1 sec, State POINT_TO_POINT,
Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
Hello due in 00:00:08
Adjacency State FULL (Hello suppressed)
Number of DBD retrans during last exchange 0
Index 2/2, retransmission queue length 0, number of retransmission 1
First 0(0)/0(0) Next 0(0)/0(0)
Last retransmission scan length is 1, maximum is 1
Last retransmission scan time is 0 msec, maximum is 0 msec

```

Enable OSPF

This task explains how to perform the minimum OSPF configuration on your router that is to enable an OSPF process with a router ID, configure a backbone or nonbackbone area, and then assign one or more interfaces on which OSPF runs.

Before you begin

Although you can configure OSPF before you configure an IP address, no OSPF routing occurs until at least one IP address is configured.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*

- **router ospfv3** *process-name*
3. **router-id** { *router-id* }
 4. **area** *area-id*
 5. **interface** *type interface-path-id*
 6. Repeat Step 5 for each interface that uses OSPF.
 7. **log adjacency changes** [**detail**] [**enable** | **disable**]
 8. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
Enters mode.
```

Step 2 Do one of the following:

- **router ospf** *process-name*
- **router ospfv3** *process-name*

Example:

```
RP/0/(config)# router ospf 1
or
```

```
RP/0/(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }

Example:

```
RP/0/(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IP address as the router ID.

Step 4 **area** *area-id*

Example:

```
RP/0/(config-ospf)# area 0
```

Enters area configuration mode and configures an area for the OSPF process.

- Backbone areas have an area ID of 0.
- Nonbackbone areas have a nonzero area ID.
- The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 5 `interface` *type interface-path-id*

Example:

```
RP/0/(config-ospf-ar)# interface TenGigE 0/1/0/3
```

Enters interface configuration mode and associates one or more interfaces for the area configured in Step 4.

Step 6 Repeat Step 5 for each interface that uses OSPF.

—

Step 7 `log adjacency changes` [**detail**] [**enable** | **disable**]

Example:

```
RP/0/(config-ospf-ar-if)# log adjacency changes detail
```

(Optional) Requests notification of neighbor changes.

- By default, this feature is enabled.
- The messages generated by neighbor changes are considered notifications, which are categorized as severity Level 5 in the **logging console** command. The **logging console** command controls which severity level of messages are sent to the console. By default, all severity level messages are sent.

Step 8 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Enable OSPF: Example

OSPF areas must be explicitly configured, and interfaces configured under the area configuration mode are explicitly bound to that area. In this example, interface 10.1.2.0/24 is bound to area 0 and interface 10.1.3.0/24 is bound to area 1.

```
interface TenGigE 0/3/0/0
 ip address 10.1.2.1 255.255.255.0
 negotiation auto
```

```

!
interface TenGigE 0/3/0/1
 ip address 10.1.3.1 255.255.255.0
 negotiation auto
!
router ospf 1
 router-id 10.2.3.4
 area 0
  interface TenGigE 0/3/0/0
!
 area 1
  interface TenGigE 0/3/0/1
!
!

```

Verify OSPF Configuration and Operation

This task explains how to verify the configuration and operation of OSPF.

SUMMARY STEPS

1. **show** { ospf | ospfv3 } [*process-name*]
2. **show** { ospf | ospfv3 } [*process-name*] **border-routers** [*router-id*]
3. **show** { ospf | ospfv3 } [*process-name*] **database**
4. **show** { ospf | ospfv3 } [*process-name*] [*area-id*] **flood-list interface** *type interface-path-id*
5. **show** { ospf | ospfv3 } [*process-name*] [**vrf** *vrf-name*] [*area-id*] **interface** [*type interface-path-id*]
6. **show** { ospf | ospfv3 } [*process-name*] [*area-id*] **neighbor** [*type interface-path-id*] [*neighbor-id*] [**detail**]
7. **clear** { ospf | ospfv3 } [*process-name*] **process**
8. **clear** { ospf | ospfv3 } [*process-name*] **redistribution**
9. **clear** { ospf | ospfv3 } [*process-name*] **routes**
10. **clear** { ospf | ospfv3 } [*process-name*] **vrf** [*vrf-name* | **all**] { **process** | **redistribution** | **routes** | **statistics** [**interface** *type interface-path-id* | **message-queue** | **neighbor**] }
11. **clear** { ospf | ospfv3 } [*process-name*] **statistics** [**neighbor** [*type interface-path-id*] [*ip-address*]]

DETAILED STEPS

Step 1 **show** { ospf | ospfv3 } [*process-name*]

Example:

```
RP/0/# show ospf group1
```

(Optional) Displays general information about OSPF routing processes.

Step 2 **show** { ospf | ospfv3 } [*process-name*] **border-routers** [*router-id*]

Example:

```
RP/0/# show ospf group1 border-routers
```

(Optional) Displays the internal OSPF routing table entries to an ABR and ASBR.

Step 3 **show { ospf | ospfv3 } [process-name] database**

Example:

```
RP/0/# show ospf group2 database
```

(Optional) Displays the lists of information related to the OSPF database for a specific router.

- The various forms of this command deliver information about different OSPF LSAs.

Step 4 **show { ospf | ospfv3 } [process-name] [area-id] flood-list interface type interface-path-id**

Example:

```
RP/0/# show ospf 100 flood-list interface TenGigE 0/3/0/0
```

(Optional) Displays a list of OSPF LSAs waiting to be flooded over an interface.

Step 5 **show { ospf | ospfv3 } [process-name] [vrf vrf-name] [area-id] interface [type interface-path-id]**

Example:

```
RP/0/# show ospf 100 interface TenGigE 0/3/0/0
```

(Optional) Displays OSPF interface information.

Step 6 **show { ospf | ospfv3 } [process-name] [area-id] neighbor [type interface-path-id] [neighbor-id] [detail]**

Example:

```
RP/0/# show ospf 100 neighbor
```

(Optional) Displays OSPF neighbor information on an individual interface basis.

Step 7 **clear { ospf | ospfv3 } [process-name] process**

Example:

```
RP/0/  
/CPU0:router# clear ospf 100 process
```

(Optional) Resets an OSPF router process without stopping and restarting it.

Step 8 **clear {ospf|ospfv3} [process-name] redistribution**

Example:

```
RP/0/#clear ospf 100 redistribution
```

Clears OSPF route redistribution.

Step 9 **clear {ospf|ospfv3} [process-name] routes**

Example:

```
RP/0/#clear ospf 100 routes
```

Clears OSPF route table.

Step 10 `clear {ospf|ospfv3 [process-name] vrf [vrf-name|all] {process |redistribution|routes|statistics [interface type interface-path-id|message-queue|neighbor]}}`

Example:

```
RP/0/#clear ospf 100 vrf vrf_1 process
```

Clears OSPF route table.

Step 11 `clear { ospf | ospfv3 } [process-name] statistics [neighbor [type interface-path-id] [ip-address]]`

Example:

```
RP/0/# clear ospf 100 statistics
```

(Optional) Clears the OSPF statistics of neighbor state transitions.

Control Frequency That Same LSA Is Originated or Accepted for OSPF

This task explains how to tune the convergence time of OSPF routes in the routing table when many LSAs need to be flooded in a very short time interval.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - `router ospf process-name`
 - `router ospfv3 process-name`
3. `router-id { router-id }`
4. Perform Step 5 or Step 6 or both to control the frequency that the same LSA is originated or accepted.
5. `timers lsa refresh seconds`
6. `timers lsa min-arrival seconds`
7. `timers lsa group-pacing seconds`
8. Use the `commit` or `end` command.

DETAILED STEPS

Step 1 `configure`

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 Do one of the following:

- **router ospf** *process-name*
- **router ospfv3** *process-name*

Example:

```
RP/0/:router(config)# router ospf 1
```

or

```
RP/0/(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }

Example:

```
RP/0/(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IP address as the router ID.

Step 4 Perform Step 5 or Step 6 or both to control the frequency that the same LSA is originated or accepted.

—

Step 5 **timers lsa refresh** *seconds*

Example:

```
RP/0/(config-ospf)# timers lsa refresh 1800
```

Sets how often self-originated LSAs should be refreshed, in seconds.

- The default is 1800 seconds for both OSPF and OSPFv3.

Step 6 **timers lsa min-arrival** *seconds*

Example:

```
RP/0/(config-ospf)# timers lsa min-arrival 2
```

Limits the frequency that new processes of any particular OSPF Version 2 LSA can be accepted during flooding.

- The default is 1 second.

Step 7 **timers lsa group-pacing** *seconds*

Example:

```
RP/0/  
/CPU0:router(config-ospf)# timers lsa group-pacing 1000
```

Changes the interval at which OSPF link-state LSAs are collected into a group for flooding.

- The default is 240 seconds.

Step 8 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Create Virtual Link

This task explains how to create a virtual link to your backbone (area 0) and apply MD5 authentication. You must perform the steps described on both ABRs, one at each end of the virtual link.



Note After you explicitly configure area parameter values, they are inherited by all interfaces bound to that area—unless you override the values and configure them explicitly for the interface.

Before you begin

The following prerequisites must be met before creating a virtual link with MD5 authentication to area 0:

- You must have the router ID of the neighbor router at the opposite end of the link to configure the local router. You can execute the **show ospf** or **show ospfv3** command on the remote router to get its router ID.
- For a virtual link to be successful, you need a stable router ID at each end of the virtual link. You do not want them to be subject to change, which could happen if they are assigned by default. . Therefore, we recommend that you perform one of the following tasks before configuring a virtual link:
 - Use the **router-id** command to set the router ID. This strategy is preferable.
 - Configure a loopback interface so that the router has a stable router ID.
- Before configuring your virtual link for OSPF Version 2, you must decide whether to configure plain text authentication, MD5 authentication, or no authentication (which is the default). Your decision determines whether you need to perform additional tasks related to authentication.

SUMMARY STEPS

1. Do one of the following:

- **show ospf** [*process-name*]
 - **show ospfv3** [*process-name*]
2. **configure**
 3. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
 4. **router-id** { *router-id* }
 5. **area** *area-id*
 6. **virtual-link** *router-id*
 7. **authentication message-digest**
 8. **message-digest-key** *key-id* **md5** { *key* | **clear** *key* | **encrypted** *key* }
 9. Repeat all of the steps in this task on the ABR that is at the other end of the virtual link. Specify the same key ID and key that you specified for the virtual link on this router.
 10. Use the **commit** or **end** command.
 11. Do one of the following:
 - **show ospf** [*process-name*] [*area-id*] **virtual-links**
 - **show ospfv3** [*process-name*] **virtual-links**

DETAILED STEPS

Step 1

Do one of the following:

- **show ospf** [*process-name*]
- **show ospfv3** [*process-name*]

Example:

```
RP/0//CPU0:router# show ospf
```

or

```
RP/0//CPU0:router# show ospfv3
```

(Optional) Displays general information about OSPF routing processes.

- The output displays the router ID of the local router. You need this router ID to configure the other end of the link.

Step 2

configure

Example:

```
RP/0/# configure
```

Enters mode.

Step 3

Do one of the following:

- **router ospf** *process-name*
- **router ospfv3** *process-name*

Example:

```
RP/0//CPU0:router(config)# router ospf 1
```

or

```
RP/0//CPU0:router(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 4 **router-id** { *router-id* }**Example:**

```
RP/0//CPU0:router(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IPv4 address as the router ID.

Step 5 **area** *area-id***Example:**

```
RP/0//CPU0:router(config-ospf)# area 1
```

Enters area configuration mode and configures a nonbackbone area for the OSPF process.

- The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 6 **virtual-link** *router-id***Example:**

```
RRP/0//CPU0:router(config-ospf-ar)# virtual-link 10.3.4.5
```

Defines an OSPF virtual link.

- See .

Step 7 **authentication message-digest****Example:**

```
RP/0//CPU0:router(config-ospf-ar-vl)#authentication message-digest
```

Selects MD5 authentication for this virtual link.

Step 8 **message-digest-key** *key-id* **md5** { *key* | **clear** *key* | **encrypted** *key* }**Example:**

```
RP/0//CPU0:router(config-ospf-ar-vl)#message-digest-key 4 md5 yourkey
```

Defines an OSPF virtual link.

- See [to understand a virtual link](#).
- The *key-id* argument is a number in the range from 1 to 255. The *key* argument is an alphanumeric string of up to 16 characters. The routers at both ends of the virtual link must have the same key identifier and key to be able to route OSPF traffic.
- The **authentication-key** *key* command is not supported for OSPFv3.
- Once the key is encrypted it must remain encrypted.

Step 9 Repeat all of the steps in this task on the ABR that is at the other end of the virtual link. Specify the same key ID and key that you specified for the virtual link on this router.

—

Step 10 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 11 Do one of the following:

- **show ospf** [*process-name*] [*area-id*] **virtual-links**
- **show ospfv3** [*process-name*] **virtual-links**

Example:

```
RP/0//CPU0:router# show ospf 1 2 virtual-links
```

or

```
RP/0//CPU0:router# show ospfv3 1 virtual-links
```

(Optional) Displays the parameters and the current state of OSPF virtual links.

ABR 1 Configuration

ABR 2 Configuration

Creating an OSPFv3 virtual link

In the following example, the **show ospfv3 virtual links** command verifies that the OSPF_VL0 virtual link to the OSPFv3 neighbor is up, the ID of the virtual link interface is 2, and the IPv6 address of the virtual link endpoint is 2003:3000::1.

show ospfv3 virtual-links

```

Virtual Links for OSPFv3 1

Virtual Link OSPF_VL0 to router 10.0.0.3 is up
  Interface ID 2, IPv6 address 2003:3000::1
  Run as demand circuit
  DoNotAge LSA allowed.
  Transit area 0.1.20.255, via interface TenGigE 0/1/0/1 Cost of using 2
  Transmit Delay is 5 sec,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:02
  Adjacency State FULL (Hello suppressed)
  Index 0/2/3, retransmission queue length 0, number of retransmission 1
  First 0(0)/0(0)/0(0) Next 0(0)/0(0)/0(0)
  Last retransmission scan length is 1, maximum is 1
  Last retransmission scan time is 0 msec, maximum is 0 msec

Check for lines:
Virtual Link OSPF_VL0 to router 10.0.0.3 is up
  Adjacency State FULL (Hello suppressed)

State is up and Adjacency State is FULL

```

This example shows how to set up a virtual link to connect the backbone through area 1 for the OSPFv3 topology that consists of areas 0 and 1 and virtual links 10.0.0.217 and 10.0.0.212:

```

router ospfv3 1
router-id 10.0.0.217
area 0
  interface TenGigE 0/2/0/1
area 1
  virtual-link 10.0.0.212
  interface TenGigE 0/2/0/0

router ospfv3 1
router-id 10.0.0.212
area 0
  interface TenGigE 0/3/0/1
area 1
  virtual-link 10.0.0.217
  interface TenGigE 0/2/0/0

```

In this example, all interfaces on router ABR1 use MD5 authentication:

```

router ospf ABR1
router-id 10.10.10.10
authentication message-digest
message-digest-key 100 md5 0 cisco
area 0
  interface TenGigE 0/2/0/1
  interface TenGigE 0/3/0/0
area 1
  interface TenGigE 0/2/0/0
  virtual-link 10.10.5.5
!
!

```

In this example, only area 1 interfaces on router ABR3 use MD5 authentication:

```
router ospf ABR2
router-id 10.10.5.5
area 0
area 1
  authentication message-digest
  message-digest-key 100 md5 0 cisco
  interface TenGigE 0/9/0/1
  virtual-link 10.10.10.10
area 3
  interface Loopback 0
  interface TenGigE 0/9/0/0
!
```

Summarize Subnetwork LSAs on OSPF ABR

If you configured two or more subnetworks when you assigned your IP addresses to your interfaces, you might want the software to summarize (aggregate) into a single LSA all of the subnetworks that the local area advertises to another area. Such summarization would reduce the number of LSAs and thereby conserve network resources. This summarization is known as interarea route summarization. It applies to routes from within the autonomous system. It does not apply to external routes injected into OSPF by way of redistribution.

This task configures OSPF to summarize subnetworks into one LSA, by specifying that all subnetworks that fall into a range are advertised together. This task is performed on an ABR only.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **area** *area-id*
5. Do one of the following:
 - **range** *ip-address mask* [**advertise** | **not-advertise**]
 - **range** *ipv6-prefix / prefix-length* [**advertise** | **not-advertise**]
6. **interface** *type interface-path-id*
7. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 Do one of the following:

- **router ospf** *process-name*
- **router ospfv3** *process-name*

Example:

```
RP/0/(config)# router ospf 1
```

or

```
RP/0/(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }

Example:

```
RP/0/(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IPv4 address as the router ID.

Step 4 **area** *area-id*

Example:

```
RP/0/(config-ospf)# area
```

Enters area configuration mode and configures a nonbackbone area for the OSPF process.

- The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 5 Do one of the following:

- **range** *ip-address mask* [**advertise** | **not-advertise**]
- **range** *ipv6-prefix / prefix-length* [**advertise** | **not-advertise**]

Example:

```
RP/0/(config-ospf-ar)# range 192.168.0.0 255.255.0.0 advertise
```

or

```
RP/0/(config-ospf-ar)# range 4004:f000::/32 advertise
```

Consolidates and summarizes OSPF routes at an area boundary.

- The **advertise** keyword causes the software to advertise the address range of subnetworks in a Type 3 summary LSA.

- The **not-advertise** keyword causes the software to suppress the Type 3 summary LSA, and the subnetworks in the range remain hidden from other areas.
- In the first example, all subnetworks for network 192.168.0.0 are summarized and advertised by the ABR into areas outside the backbone.
- In the second example, two or more IPv4 interfaces are covered by a 192.x.x network.

Step 6 `interface type interface-path-id`

Example:

```
RP/0/(config-ospf-ar)# interface TenGigE 0/0/0/0
```

Enters interface configuration mode and associates one or more interfaces to the area.

Step 7 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Example

The following example shows the prefix range 2300::/16 summarized from area 1 into the backbone:

```
router ospfv3 1
router-id 192.168.0.217
area 0
interface TenGigE 0/0/0/0
area 1
range 2300::/16
interface TenGigE 0/0/0/0
```

Graceful Shutdown for OSPFv3

The OSPFv3 Graceful Shutdown feature preserves the data plane capability in these circumstances:

- RP failure resulting in a switch-over to the backup processor
- Planned OSPFv3 process restart, such as a restart resulting from a software upgrade or downgrade
- Unplanned OSPFv3 process restart, such as a restart resulting from a process crash

In addition, OSPFv3 will unilaterally shutdown and enter the exited state when a critical memory event, indicating the processor is critically low on available memory, is received from the sysmon watch dog process.

This feature supports nonstop data forwarding on established routes while the OSPFv3 routing protocol restarts. Therefore, this feature enhances high availability of IPv6 forwarding.

Enable Nonstop Routing for OSPFv2

This optional task describes how to enable nonstop routing (NSR) for OSPFv2 process. NSR is disabled by default. When NSR is enabled, OSPF process on the active RP synchronizes all necessary data and states with the OSPF process on the standby RP. When the switchover happens, OSPF process on the newly active RP has all the necessary data and states to continue running and does not require any help from its neighbors.

SUMMARY STEPS

1. **configure**
2. **router ospf** *instance-id*
3. **nsr**
4. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure****Example:**

```
RP/0/# configure
```

Enters mode.

Step 2 **router ospf** *instance-id***Example:**

```
RP/0/(config)# router ospf isp
```

Enables OSPF routing for the specified routing process, and places the router in router configuration mode. In this example, the OSPF instance is called `isp`.

Step 3 **nsr****Example:**

```
RP/0/(config-ospf)# nsr
```

Enables NSR for the OSPFv2 process.

Step 4 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.

- **Cancel**—Remains in the configuration session, without committing the configuration changes.

Warm Standby and Nonstop Routing for OSPF Version 3

This feature helps OSPFv3 to initialize itself prior to Fail over (FO) and be ready to function before the failure occurs. It reduces the downtime during switchover. By default, the router sends hello packets every 40 seconds.

With warm standby process for each OSPF process running on the Active Route Processor, the corresponding OSPF process must start on the Standby RP. There are no changes in configuration for this feature.

Warm-Standby is always enabled. This is an advantage for the systems running OSPFv3 as their IGP when they do RP failover.

Enable Nonstop Routing for OSPFv3

This task describes how to enable nonstop routing (NSR) for OSPFv3 process. NSR is disabled by default. When NSR is enabled, OSPF process on the active RP synchronizes all necessary data and states with the OSPF process on the standby RP. When the switchover happens, OSPF process on the newly active RP has all the necessary data and states to continue running and does not require any help from its neighbors.

SUMMARY STEPS

1. **configure**
2. **router ospfv3** *instance-id*
3. **nsr**
4. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure****Example:**

```
RP/0/# configure
```

Enters mode.

Step 2 **router ospfv3** *instance-id***Example:**

```
RP/0/(config)# router ospfv3 isp
```

Enables OSPF routing for the specified routing process, and places the router in router configuration mode. In this example, the OSPF instance is called isp.

Step 3 **nsr****Example:**

```
RP/0/(config-ospfv3)# nsr
```

Enables NSR for the OSPFv3 process.

Step 4 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

OSPFv2OSPF SPF Prefix Prioritization

The OSPFv2 OSPF SPF Prefix Prioritization feature enables an administrator to converge, in a faster mode, important prefixes during route installation.

When a large number of prefixes must be installed in the Routing Information Base (RIB) and the Forwarding Information Base (FIB), the update duration between the first and last prefix, during SPF, can be significant.

In networks where time-sensitive traffic (for example, VoIP) may transit to the same router along with other traffic flows, it is important to prioritize RIB and FIB updates during SPF for these time-sensitive prefixes.

The OSPFv2OSPF SPF Prefix Prioritization feature provides the administrator with the ability to prioritize important prefixes to be installed, into the RIB during SPF calculations. Important prefixes converge faster among prefixes of the same route type per area. Before RIB and FIB installation, routes and prefixes are assigned to various priority batch queues in the OSPF local RIB, based on specified route policy. The RIB priority batch queues are classified as "critical," "high," "medium," and "low," in the order of decreasing priority.

When enabled, prefix alters the sequence of updating the RIB with this prefix priority:

Critical > High > Medium > Low

As soon as prefix priority is configured, /32 prefixes are no longer preferred by default; they are placed in the low-priority queue, if they are not matched with higher-priority policies. Route policies must be devised to retain /32s in the higher-priority queues (high-priority or medium-priority queues).

Priority is specified using route policy, which can be matched based on IP addresses or route tags. During SPF, a prefix is checked against the specified route policy and is assigned to the appropriate RIB batch priority queue.

These are examples of this scenario:

- If only high-priority route policy is specified, and no route policy is configured for a medium priority:
 - Permitted prefixes are assigned to a high-priority queue.
 - Unmatched prefixes, including /32s, are placed in a low-priority queue.
- If both high-priority and medium-priority route policies are specified, and no maps are specified for critical priority:

- Permitted prefixes matching high-priority route policy are assigned to a high-priority queue.
 - Permitted prefixes matching medium-priority route policy are placed in a medium-priority queue.
 - Unmatched prefixes, including /32s, are moved to a low-priority queue.
- If both critical-priority and high-priority route policies are specified, and no maps are specified for medium priority:
 - Permitted prefixes matching critical-priority route policy are assigned to a critical-priority queue.
 - Permitted prefixes matching high-priority route policy are assigned to a high-priority queue.
 - Unmatched prefixes, including /32s, are placed in a low-priority queue.
 - If only medium-priority route policy is specified and no maps are specified for high priority or critical priority:
 - Permitted prefixes matching medium-priority route policy are assigned to a medium-priority queue.
 - Unmatched prefixes, including /32s, are placed in a low-priority queue.

Use the **[no] spf prefix-priority route-policy** *rpl* command to prioritize OSPFv2/OSPF prefix installation into the global RIB during SPF.

SPF prefix prioritization is disabled by default. In disabled mode, /32 prefixes are installed into the global RIB, before other prefixes. If SPF prioritization is enabled, routes are matched against the route-policy criteria and are assigned to the appropriate priority queue based on the SPF priority set. Unmatched prefixes, including /32s, are placed in the low-priority queue.

If all /32s are desired in the high-priority queue or medium-priority queue, configure this single route map:

```
prefix-set ospf-medium-prefixes
0.0.0.0/0 ge 32
end-set
```

Configuring OSPFv2 SPF Prefix Prioritization: Example

This example shows how to configure /32 prefixes as medium-priority, in general, in addition to placing some /32 and /24 prefixes in critical-priority and high-priority queues:

```
prefix-set ospf-critical-prefixes
192.41.5.41/32,
11.1.3.0/24,
192.168.0.44/32
end-set
!
prefix-set ospf-high-prefixes
44.4.10.0/24,
192.41.4.41/32,
41.4.41.41/32
end-set
!
prefix-set ospf-medium-prefixes
0.0.0.0/0 ge 32
end-set
```

```

!
route-policy ospf-priority
  if destination in ospf-high-prefixes then
    set spf-priority high
  else
    if destination in ospf-critical-prefixes then
      set spf-priority critical
    else
      if destination in ospf-medium-prefixes then
        set spf-priority medium
      endif
    endif
  endif
end-policy

```

OSPFv2

```

router ospf 1
  spf prefix-priority route-policy ospf-priority
  area 0
    interface GigabitEthernet0/3/0/0
    !
  area 3
    interface GigabitEthernet0/2/0/0
    !
  area 8
    interface GigabitEthernet0/2/0/0.590

```

OSPFv3 Timers Update

The Open Shortest Path First version 3 (OSPFv3) timers link-state advertisements (LSAs) and shortest path first (SPF) throttle default values are updated to:

- **timers throttle lsa all**—*start-interval*: 50 milliseconds and *hold-interval*: 200 milliseconds
- **timers throttle spf**—*spf-start*: 50 milliseconds, *spf-hold*: 200 milliseconds, *spf-max-wait*: 5000 milliseconds

OSPF Link-State Database Overload Protection

The OSPF Link-State Database Overload Protection feature allows you to protect the OSPF routing process from the large number of received link-state advertisements (LSAs) that can result from a misconfiguration on another router in the OSPF domain (for example, the redistribution of a large number of IP prefixes to OSPF). In this feature, the router monitors the count of all nonself-generated LSAs it receives. When the total number of LSAs reaches the value set as a threshold in the configuration, the router logs a warning message. When the total LSAs exceed the max number in configuration, the router stops accepting new LSAs. Therefore, the OSPF Link-State Database Overload Protection feature allows you to limit the number of nonself-generated LSAs for a given OSPF process. Excessive LSAs generated by other routers in the OSPF domain can substantially drain the CPU and memory resources of the router.

If the count of received LSAs is higher than the configured max number after one minute, the OSPF process disables all adjacencies in the given context and clears the OSPF database. This state is called the ignore state. In this state, the router ignores all OSPF packets received on all interfaces belonging to the OSPF instance and stops OSPF packet generation on its interfaces. The OSPF process remains in ignore state for the duration of the configured ignore time. When the ignore time expires, the OSPF process returns to a normal operation state and starts building adjacencies on all its interfaces.

To prevent the OSPF instance from endlessly oscillating between the normal state and ignore state, as a result of the LSA counts immediately exceeding the max number again after it returns from the ignore state, the OSPF instance keeps a count of how many times it has been in ignore state. This counter is called the ignore count. If the ignore count exceeds its configured value, the OSPF instance remains in ignore state permanently.

To return the OSPF instance to its normal state, use the **clear IP OSPF process** command. The ignore count is reset to zero if the LSA count doesn't exceed the max number again during the time configured by the reset-time keyword.



Note There are no default values for the **max-lsa** command. The router considers the max-lsa limits only if it's available in the configuration. After configuring the OSPF Link-State Database Overload Protection feature, the default values for other OSPF parameters are as follows:

- Default threshold percentage to log warning: 75%.
- Default ignore count value: 5
- Default ignore time in minutes: 5 minutes
- Default time to reset ignore count: 10 minutes

In the event of LSA count reaches or exceeds the configured threshold value, router displays the following logs:

```
Router:Oct 25 11:42:49.756 IST: ospf[1046]: %ROUTING-OSPF-4-MAX_LSA_THR : Threshold for
maximum number of non self-generated LSAs has been reached "default" - 9000 LSAs
Router:Oct 25 11:42:49.756 IST: ospf[1046]: %ROUTING-OSPF-4-MAX_LSA : Maximum number of non
self-generated LSAs has been exceeded "default" - 12001 LSAs

Router:Oct 25 11:42:49.756 IST: ospf[1046]: %ROUTING-OSPF-4-MAX_LSA_THR : Threshold for
maximum number of non self-generated LSAs has been reached "V1" - 750 LSAs
Router:Oct 25 11:42:49.756 IST: ospf[1046]: %ROUTING-OSPF-4-MAX_LSA : Maximum number of non
self-generated LSAs has been exceeded "V1" - 1001 LSA
```

Configuration Example

The following example shows how to configure the OSPF instance to accept 12000 nonself-generated LSAs in the global routing table and 1000 nonself-generated LSAs in VRF V1:

```
Router# configure
Router(config)# router ospf 0
Router(config-ospf)# max-lsa 12000
Router(config-ospf)# vrf V1
Router(config-ospf)# max-lsa 1000
```

Running Configuration

The following example shows how to display the current status of the OSPF instance:

```
Router# show ospf 0
Routing Process "ospf 0" with ID 10.0.0.2
NSR (Non-stop routing) is Disabled
Supports only single TOS(TOS0) routes
Supports opaque LSA
It is an area border router
Maximum number of non self-generated LSA allowed 12000
  Current number of non self-generated LSA 1
  Threshold for warning message 75%
  Ignore-time 5 minutes, reset-time 10 minutes
  Ignore-count allowed 5, current ignore-count 0

Router# show ospf vrf V1
VRF V1 active in Routing Process "ospf 0" with ID 10.0.0.2
Role: Primary Active
NSR (Non-stop routing) is Enabled
Supports only single TOS(TOS0) routes
Supports opaque LSA
It is an area border router
Maximum number of non self-generated LSA allowed 1000
  Current number of non self-generated LSA 1
  Threshold for warning message 75%
  Ignore-time 5 minutes, reset-time 10 minutes
  Ignore-count allowed 5, current ignore-count 0
Router is not originating router-LSAs with maximum metric
```



CHAPTER 3

Implementing and Monitoring RIB

Routing Information Base (RIB) is a distributed collection of information about routing connectivity among all nodes of a network. Each router maintains a RIB containing the routing information for that router. RIB stores the best routes from all routing protocols that are running on the system.

Each routing protocol selects its own set of best routes and installs those routes and their attributes in RIB. RIB stores these routes and selects the best ones from among all routing protocols. Those routes are downloaded to the line cards for use in forwarding packets. The acronym RIB is used both to refer to RIB processes and the collection of route data contained within RIB. Within a protocol, routes are selected based on the metrics in use by that protocol. A protocol downloads its best routes (lowest or tied metric) to RIB. RIB selects the best overall route by comparing the administrative distance of the associated protocol.

This module describes how to implement and monitor RIB on your network.



Tip You can programmatically configure RIB for Border Gateway Protocol (BGP) and retrieve operational data using `openconfig-rib-bgp.yang` OpenConfig data model. To get started with using data models, see the .

- [Prerequisites for Implementing RIB, on page 199](#)
- [Information About RIB Configuration, on page 200](#)
- [How to Deploy and Monitor RIB, on page 203](#)
- [BGP-RIB Feedback Mechanism for Update Generation, on page 205](#)
- [Configuration Examples for RIB Monitoring, on page 206](#)

Prerequisites for Implementing RIB

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- RIB is distributed with the base Cisco IOS XR software; as such, it does not have any special requirements for installation. The following are the requirements for base software installation:
 - Router
 - Cisco IOS XR software
 - Base package

Information About RIB Configuration

To implement the Cisco RIB feature, you must understand the following concepts:

Overview of RIB

Each routing protocol selects its own set of best routes and installs those routes and their attributes in RIB. RIB stores these routes and selects the best ones from among all routing protocols. Those routes are downloaded to the line cards for use in forwarding packets. The acronym RIB is used both to refer to RIB processes and the collection of route data contained within RIB.

Within a protocol, routes are selected based on the metrics in use by that protocol. A protocol downloads its best routes (lowest or tied metric) to RIB. RIB selects the best overall route by comparing the administrative distance of the associated protocol.

RIB Data Structures in BGP and Other Protocols

RIB uses processes and maintains data structures distinct from other routing applications, such as Border Gateway Protocol (BGP) and other unicast routing protocols. However, these routing protocols use internal data structures similar to what RIB uses, and may internally refer to the data structures as a RIB. For example, BGP routes are stored in the BGP RIB (BRIB). RIB processes are not responsible for the BRIB, which are handled by BGP.

The table used by the line cards and RP to forward packets is called the Forwarding Information Base (FIB). RIB processes do not build the FIBs. Instead, RIB downloads the set of selected best routes to the FIB processes, by the Bulk Content Downloader (BCDL) process, onto each line card. FIBs are then constructed.

RIB Administrative Distance

Forwarding is done based on the longest prefix match. If you are forwarding a packet destined to 10.0.2.1, you prefer 10.0.2.0/24 over 10.0.0.0/16 because the mask /24 is longer (and more specific) than a /16.

Routes from different protocols that have the same prefix and length are chosen based on administrative distance. For instance, the Open Shortest Path First (OSPF) protocol has an administrative distance of 110, and the Intermediate System-to-Intermediate System (IS-IS) protocol has an administrative distance of 115. If IS-IS and OSPF both download 10.0.1.0/24 to RIB, RIB would prefer the OSPF route because OSPF has a lower administrative distance. Administrative distance is used only to choose between multiple routes of the same length.

This table lists default administrative distances for the common protocols.

Table 3: Default Administrative Distances

Protocol	Administrative Distance Default
Connected or local routes	0
Static routes	1
External BGP routes	20

Protocol	Administrative Distance Default
OSPF routes	110
IS-IS routes	115
Internal BGP routes	200

The administrative distance for some routing protocols (for instance IS-IS, OSPF, and BGP) can be changed. See the protocol-specific documentation for the proper method to change the administrative distance of that protocol.



Note Changing the administrative distance of a protocol on some but not all routers can lead to routing loops and other undesirable behavior. Doing so is not recommended.

RIB Support for IPv4

In Cisco IOS XR software, RIB tables support unicast routing.

The default routing tables for Cisco IOS XR software RIB are the unicast RIB tables for IPv4 routing.

RIB processes `ipv4_rib` and `ipv6_rib` run on the RP card. If process placement functionality is available and supported by multiple RPs in the router, RIB processes can be placed on any available node.

RIB Statistics

RIB supports statistics for messages (requests) flowing between the RIB and its clients. Protocol clients send messages to the RIB (for example, route add, route delete, and next-hop register, and so on). RIB also sends messages (for example, redistribute routes, advertisements, next-hop notifications, and so on). These statistics are used to gather information about what messages have been sent and the number of messages that have been sent. These statistics provide counters for the various messages that flow between the RIB server and its clients. The statistics are displayed using the **show rib statistics** command.

RIB maintains counters for all requests sent from a client including:

- Route operations
- Table registrations
- Next-hop registrations
- Redistribution registrations
- Attribute registrations
- Synchronization completion

RIB also maintains counters for all requests sent by the RIB. The configuration will disable the RIB next-hop dampening feature. As a result, RIB notifies client immediately when a next hop that client registered for is resolved or unresolved.

RIB also maintains the results of the requests.

RIB Quarantining

RIB quarantining solves the problem in the interaction between routing protocols and the RIB. The problem is a persistent oscillation between the RIB and routing protocols that occurs when a route is continuously inserted and then withdrawn from the RIB, resulting in a spike in CPU use until the problem is resolved. If there is no damping on the oscillation, then both the protocol process and the RIB process have high CPU use, affecting the rest of the system as well as blocking out other protocol and RIB operations. This problem occurs when a particular combination of routes is received and installed in the RIB. This problem typically happens as a result of a network misconfiguration. However, because the misconfiguration is across the network, it is not possible to detect the problem at configuration time on any single router.

The quarantining mechanism detects mutually recursive routes and quarantines the last route that completes the mutual recursion. The quarantined route is periodically evaluated to see if the mutual recursion has gone away. If the recursion still exists, the route remains quarantined. If the recursion has gone away, the route is released from its quarantine.

The following steps are used to quarantine a route:

1. RIB detects when a particular problematic path is installed.
2. RIB sends a notification to the protocol that installed the path.
3. When the protocol receives the quarantine notification about the problem route, it marks the route as being “quarantined.” If it is a BGP route, BGP does not advertise reachability for the route to its neighbors.
4. Periodically, RIB tests all its quarantined paths to see if they can now safely be installed (moved from quarantined to "Ok to use" state). A notification is sent to the protocol to indicate that the path is now safe to use.

Route Consistency Checker

The Route Consistency Checker (RCC) is a command-line tool that can be used to verify consistency between control plane and data plane route in IOS XR software.

Routers in production networks may end up in a state where the forwarding information does not match the control plane information. Possible causes of this include fabric or transport failures between the Route Processor (RP) and the line cards (LCs), or issues with the Forwarding Information Base (FIB). RCC can be used to identify and provide detailed information about resultant inconsistencies between the control plane and data plane. This information can be used to further investigate and diagnose the cause of forwarding problems and traffic loss.

RCC can be run in two modes. It can be triggered from EXEC mode as an on-demand, one-time scan (On-demand Scan), or be configured to run at defined intervals in the background during normal router operation (Background Scan). RCC compares the Routing Information Base (RIB) against the Forwarding Information Base (FIB). When an inconsistency is detected, RCC output will identify the specific route and identify the type of inconsistency detected as well as provide additional data that will assist with further troubleshooting.

RCC runs on the Route Processor. FIB checks for errors on the line card and forwards first the 20 error reports to RCC. RCC receives error reports from all nodes, summarizes them (checks for exact match), and adds it to two queues, soft or hard. Each queue has a limit of 1000 error reports and there is no prioritization in the queue. RCC logs the same errors (exact match) from different nodes as one error. RCC compares the errors based on prefix, version number, type of error, etc.

On-demand Scan

In On-demand Scan, user requests scan through the command line interface on a particular prefix in a particular table or all the prefixes in the table. The scan is run immediately and the results are published right away. RCC performs on-demand scan per VRF.

Background Scan

In Background Scan, user configures the scan that is then left to run in the background. The configuration consists of the time period for the periodic scan. This scan can be configured on either a single table or multiple tables. RCC performs background scan either for default or other VRFs.

How to Deploy and Monitor RIB

To deploy and monitor RIB, you must understand the following concepts:

Verifying RIB Configuration Using the Routing Table

Perform this task to verify the RIB configuration to ensure that RIB is running on the RP and functioning properly by checking the routing table summary and details.

Procedure

	Command or Action	Purpose
Step 1	show route [ipv4 ipv6] [unicast] Example: <pre>Router# show route summary</pre>	Displays route summary information about the specified routing table. <ul style="list-style-type: none"> The default table summarized is the IPv4 unicast routing table.
Step 2	show route [ipv4 ipv6] [unicast] Example: <pre>Router# show route ipv4 unicast</pre>	Displays more detailed route information about the specified routing table. <ul style="list-style-type: none"> This command is usually issued with an IP address or other optional filters to limit its display. Otherwise, it displays all routes from the default IPv4 unicast routing table, which can result in an extensive list, depending on the configuration of the network.

Verifying Networking and Routing Problems

Perform this task to verify the operation of routes between nodes.

Procedure

	Command or Action	Purpose
Step 1	<p>show route [<i>afi-all</i> <i>ipv4</i> <i>ipv6</i>] [<i>unicast</i> <i>multicast</i> <i>safi-all</i>] [<i>protocol</i> [<i>instance</i>] <i>ip-address mask</i>] [<i>standby</i>] [<i>detail</i>]</p> <p>Example:</p> <pre>Router# show route ipv4 unicast 192.168.1.11/8</pre>	Displays the current routes in RIB.
Step 2	<p>show route [<i>afi-all</i> <i>ipv4</i> <i>ipv6</i>] [<i>unicast</i> <i>multicast</i> <i>safi-all</i>] <i>backup</i> [<i>ip-address</i>] [<i>standby</i>]</p> <p>Example:</p> <pre>Router# show route ipv4 unicast backup 192.168.1.11/8</pre>	Displays backup routes in RIB.
Step 3	<p>show route [<i>ipv4</i> <i>ipv6</i>] [<i>unicast</i> <i>multicast</i> <i>safi-all</i>] <i>best-local ip-address</i> [<i>standby</i>]</p> <p>Example:</p> <pre>Router# show route ipv4 unicast best-local 192.168.1.11/8</pre>	Displays the best-local address to use for return packets from the given destination.
Step 4	<p>show route [<i>afi-all</i> <i>ipv4</i> <i>ipv6</i>] [<i>unicast</i> <i>multicast</i> <i>safi-all</i>] <i>connected</i> [<i>standby</i>]</p> <p>Example:</p> <pre>Router# show route ipv4 unicast connected</pre>	Displays the current connected routes of the routing table.
Step 5	<p>show route [<i>afi-all</i> <i>ipv4</i> <i>ipv6</i>] [<i>unicast</i> <i>multicast</i> <i>safi-all</i>] <i>local</i> [<i>interface</i>] [<i>standby</i>]</p> <p>Example:</p> <pre>Router# show route ipv4 unicast local</pre>	Displays local routes for receive entries in the routing table.
Step 6	<p>show route [<i>ipv4</i> <i>ipv6</i>] [<i>unicast</i> <i>multicast</i> <i>safi-all</i>] <i>longer-prefixes</i> { <i>ip-address mask</i> <i>ip-address / prefix-length</i> } [<i>standby</i>]</p> <p>Example:</p> <pre>Router# show route ipv4 unicast longer-prefixes 192.168.1.11/8</pre>	Displays the current routes in RIB that share a given number of bits with a given network.
Step 7	<p>show route [<i>ipv4</i> <i>ipv6</i>] [<i>unicast</i> <i>multicast</i> <i>safi-all</i>] <i>next-hop ip-address</i> [<i>standby</i>]</p> <p>Example:</p> <pre>Router# show route ipv4 unicast next-hop 192.168.1.34</pre>	Displays the next-hop gateway or host to a destination address.

Disabling RIB Next-hop Dampening

Perform this task to disable RIB next-hop dampening.

SUMMARY STEPS

1. **router rib**
2. **address-family { ipv4 | ipv6 } next-hop dampening disable**
3. Use the **commit** or **end** command.

DETAILED STEPS

	Command or Action	Purpose
Step 1	router rib Example: <pre>Router# route rib</pre>	Enters RIB configuration mode.
Step 2	address-family { ipv4 ipv6 } next-hop dampening disable Example: <pre>Router(config-rib)# address family ipv4 next-hop dampening disable</pre>	Disables next-hop dampening for IPv4 address families.
Step 3	Use the commit or end command.	commit —Saves the configuration changes and remains within the configuration session. end —Prompts user to take one of these actions: <ul style="list-style-type: none"> • Yes — Saves configuration changes and exits the configuration session. • No —Exits the configuration session without committing the configuration changes. • Cancel —Remains in the configuration session, without committing the configuration changes.

BGP-RIB Feedback Mechanism for Update Generation

The Border Gateway Protocol-Routing Information Base (BGP-RIB) feedback mechanism for update generation feature avoids premature route advertisements and subsequent packet loss in a network. This mechanism ensures that routes are installed locally, before they are advertised to a neighbor.

BGP waits for feedback from RIB indicating that the routes that BGP installed in RIB are installed in forwarding information base (FIB) before BGP sends out updates to the neighbors. RIB uses the the BCDL feedback mechanism to determine which version of the routes have been consumed by FIB, and updates the BGP with that version. BGP will send out updates of only those routes that have versions up to the version that FIB has installed. This selective update ensures that BGP does not send out premature updates resulting in attracting

traffic even before the data plane is programmed after router reload, LC OIR, or flap of a link where an alternate path is made available.

To configure BGP to wait for feedback from RIB indicating that the routes that BGP installed in RIB are installed in FIB, before BGP sends out updates to neighbors, use the **update wait-install** command in router address-family IPv4 or router address-family VPNv4 configuration mode. The **show bgp**, **show bgp neighbors**, and **show bgp process performance-statistics** commands display the information from update wait-install configuration.

Configuration Examples for RIB Monitoring

RIB is not configured separately for the Cisco IOS XR system. RIB computes connectivity of the router with other nodes in the network based on input from the routing protocols. RIB may be used to monitor and troubleshoot the connections between RIB and its clients, but it is essentially used to monitor routing connectivity between the nodes in a network. This section contains displays from the **show** commands used to monitor that activity.

Output of show route Command: Example

The following is sample output from the **show route** command when entered without an address:

```
show route

Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local

Gateway of last resort is 172.23.54.1 to network 0.0.0.0

C    10.2.210.0/24 is directly connected, 1d21h, HundredGigE 0/1/0/0
L    10.2.210.221/32 is directly connected, 1d21h, HundredGigE 0/1/0/0
C    10.6.100.0/24 is directly connected, 1d21h, Loopback1
L    10.6.200.21/32 is directly connected, 1d21h, Loopback0
S    192.168.40.0/24 [1/0] via 172.20.16.6, 1d21h
```

Output of show route backup Command: Example

The following is sample output from the **show route backup** command:

```
show route backup

Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local

S    172.73.51.0/24 is directly connected, 2d20h, FourHundredGigE 0/0/0/1
```

```
Backup O E2 [110/1] via 10.12.12.2, FourHundredGigE 0/0/0/2
```

Output of show route best-local Command: Example

The following is sample output from the **show route best-local** command:

```
show route best-local 10.12.12.1

Routing entry for 10.12.12.1/32
  Known via "local", distance 0, metric 0 (connected)
  Routing Descriptor Blocks
    10.12.12.1 directly connected, via FourHundredGigE 0/0/0/0
    Route metric is 0
```

Output of show route connected Command: Example

The following is sample output from the **show route connected** command:

```
show route connected

C    10.2.210.0/24 is directly connected, 1d21h, HundredGigE 0/1/0/0
C    10.6.100.0/24 is directly connected, 1d21h, Loopback1
```

Output of show route local Command: Example

The following is sample output from the **show route local** command:

```
show route local

L    10.10.10.1/32 is directly connected, 00:14:36, Loopback0
L    10.91.36.98/32 is directly connected, 00:14:32, HundredGigE 0/1/0/0
L    172.22.12.1/32 is directly connected, 00:13:35, FourHundredGigE 0/1/0/0
L    192.168.20.2/32 is directly connected, 00:13:27, FourHundredGigE 0/0/0/0
L    10.254.254.1/32 is directly connected, 00:13:26, FourHundredGigE 0/1/0/0
```

Output of show route longer-prefixes Command: Example

The following is sample output from the **show route longer-prefixes** command:

```
show route ipv4 longer-prefixes 172.16.0.0/8

Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
        O - OSPF, IA - OSPF inter area, N1 - OSPF NSSA external type 1
        N2 - OSPF NSSA external type 2, E1 - OSPF external type 1
        E2 - OSPF external type 2, E - EGP, i - ISIS, L1 - IS-IS level-1
        L2 - IS-IS level-2, ia - IS-IS inter area
        su - IS-IS summary null, * - candidate default
        U - per-user static route, o - ODR, L - local

Gateway of last resort is 172.23.54.1 to network 0.0.0.0
S    172.16.2.0/32 is directly connected, 00:00:24, Loopback0
```

```

S    172.16.3.0/32 is directly connected, 00:00:24, Loopback0
S    172.16.4.0/32 is directly connected, 00:00:24, Loopback0
S    172.16.5.0/32 is directly connected, 00:00:24, Loopback0
S    172.16.6.0/32 is directly connected, 00:00:24, Loopback0
S    172.16.7.0/32 is directly connected, 00:00:24, Loopback0
S    172.16.8.0/32 is directly connected, 00:00:24, Loopback0
S    172.16.9.0/32 is directly connected, 00:00:24, Loopback0

```

Output of show route next-hop Command: Example

The following is sample output from the **show route resolving-next-hop** command:

```

show route resolving-next-hop 10.0.0.1
Mon Oct 21 10:14:25.073 UTC

NextHop matches 10.0.0.1/31
Known via "connected", distance 0, metric 0 (connected)
Installed Oct 21 10:11:22.460 for 00:03:02
Directly connected nexthops
  directly connected, via FourHundredGigE0/0/0/0
  Route metric is 0

```

Enabling RCC: Example

Enabling RCC Background Scan: Example

This example shows how to enable Route Consistency Checker (RCC) background scan with a period of 500 milliseconds between buffers in scans for IPv6 unicast tables:

```
rcc ipv6 unicast period 500
```

Enabling RCC On-demand Scan: Example

This example shows how to run Route Consistency Checker (RCC) on-demand scan for subnet 10.10.0.0/16 in vrf1:

```
show rcc ipv4 unicast 10.10.0.0/16 vrf vrf 1
```




CHAPTER 4

Implementing Routing Policy

A routing policy instructs the router to inspect routes, filter them, and potentially modify their attributes as they are accepted from a peer, advertised to a peer, or redistributed from one routing protocol to another.

This module describes how routing protocols make decisions to advertise, aggregate, discard, distribute, export, hold, import, redistribute and modify the routes based on configured routing policy.

The routing policy language (RPL) provides a single, straightforward language in which all routing policy needs can be expressed. RPL was designed to support large-scale routing configurations. It greatly reduces the redundancy inherent in previous routing policy configuration methods. RPL streamlines the routing policy configuration, reduces system resources required to store and process these configurations, and simplifies troubleshooting.

- [Restrictions for Implementing Routing Policy, on page 209](#)
- [Define Route Policy, on page 210](#)
- [Attach Routing Policy to BGP Neighbor, on page 211](#)
- [Modify Routing Policy Using Text Editor, on page 212](#)
- [References for Routing Policy, on page 215](#)

Restrictions for Implementing Routing Policy

These restrictions apply when working with Routing Policy Language implementation:

- Border Gateway Protocol (BGP), integrated Intermediate System-to-Intermediate System (IS-IS), or Open Shortest Path First (OSPF) must be configured in your network.
- An individual policy definition of up to 1000 statements are supported. The total number of statements within a policy can be extended to 4000 statements using hierarchical policy constructs. However, this limit is restricted with the use of **apply** statements.
- When a policy that is attached directly or indirectly to an attach point needs to be modified, a single **commit** operation cannot be performed when:
 - Removing a set or policy referred by another policy that is attached to any attach point directly or indirectly.
 - Modifying the policy to remove the reference to the same set or policy that is getting removed.

The **commit** must be performed in two steps:

1. Modify the policy to remove the reference to the policy or set and then **commit**.
 2. Remove the policy or set and **commit**.
- Per-vrf label mode is not supported for Carrier Supporting Carrier (CSC) network with internal and external BGP multipath setup.

Define Route Policy

This task explains how to define a route policy.



Note

- If you want to modify an existing routing policy using the command-line interface (CLI), you must redefine the policy by completing this task.
- Modifying the RPL scale configuration may take a long time.
- BGP may crash either due to large scale RPL configuration changes, or during consecutive RPL changes. To avoid BGP crash, wait until there are no messages in the BGP In/Out queue before committing further changes.



Tip

You can programmatically configure the route policy using `openconfig-routing-policy.yang` OpenConfig data model. To get started with using data models, see the .

Step 1 configure

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 route-policy name [parameter1 , parameter2 , . . . , parameterN]

Example:

```
RP/0/(config)# route-policy sample1
```

Enters route-policy configuration mode.

- After the route-policy has been entered, a group of commands can be entered to define the route-policy.

Step 3 end-policy

Example:

```
RP/0/(config-rpl)# end-policy
```

Ends the definition of a route policy and exits route-policy configuration mode.

Step 4 Use the **commit** or **end** command.

commit—Saves the configuration changes and remains within the configuration session.

end—Prompts user to take one of these actions:

- **Yes**—Saves configuration changes and exits the configuration session.
- **No**—Exits the configuration session without committing the configuration changes.
- **Cancel**—Remains in the configuration session, without committing the configuration changes.

Routing Policy Definition: Example

In the following example, a BGP route policy named `sample1` is defined using the **route-policy** *name* command. The policy compares the network layer reachability information (NLRI) to the elements in the prefix set `test`. If it evaluates to true, the policy performs the operations in the *then* clause. If it evaluates to false, the policy performs the operations in the *else* clause, that is, sets the MED value to 200 and adds the community 2:100 to the route. The final steps of the example commit the configuration to the router, exit configuration mode, and display the contents of route policy `sample1`.

```
configure
route-policy sample1
  if destination in test then
    drop
  else
    set med 200
    set community (2:100) additive
  endif
end-policy
end
show config running route-policy sample1
Building configuration...
route-policy sample1
  if destination in test then
    drop
  else
    set med 200
    set community (2:100) additive
  endif
end-policy
```

Attach Routing Policy to BGP Neighbor

This task explains how to attach a routing policy to a BGP neighbor.

Before you begin

A routing policy must be preconfigured and well defined prior to it being applied at an attach point. If a policy is not predefined, an error message is generated stating that the policy is not defined.

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 `router bgp as-number`**Example:**

```
RP/0/(config)# router bgp 125
```

Configures a BGP routing process and enters router configuration mode.

- The *as-number* argument identifies the autonomous system in which the router resides. Valid values are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.

Step 3 `neighbor ip-address`**Example:**

```
RP/0/(config-bgp)# neighbor 10.0.0.20
```

Specifies a neighbor IP address.

Step 4 `address-family { ipv4 unicast || ipv6 unicast | } address-family { ipv4 | ipv6 } unicast`**Example:**

```
RP/0/(config-bgp-nbr)# address-family ipv4 unicast
```

Specifies the address family.

Step 5 `route-policy policy-name { in | out }`**Example:**

```
RP/0/(config-bgp-nbr-af)# route-policy example1 in
```

Attaches the route-policy, which must be well formed and predefined.

Step 6 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Modify Routing Policy Using Text Editor

This task explains how to modify an existing routing policy using a text editor.

Step 1 `edit { route-policy | prefix-set | as-path-set | community-set | extcommunity-set { rt | soo } | policy-global | rd-set } name [nano | emacs | vim | inline { add | prepend | remove } set-element]`

Example:

```
RP/0/# edit route-policy sample1
```

Identifies the route policy, prefix set, AS path set, community set, or extended community set name to be modified.

- A copy of the route policy, prefix set, AS path set, community set, or extended community set is copied to a temporary file and the editor is launched.
- After editing with Nano, save the editor buffer and exit the editor by using the Ctrl-X keystroke.
- After editing with Emacs, save the editor buffer by using the Ctrl-X and Ctrl-S keystrokes. To save and exit the editor, use the Ctrl-X and Ctrl-C keystrokes.
- After editing with Vim, to write to a current file and exit, use the :wq or :x or ZZ keystrokes. To quit and confirm, use the :q keystrokes. To quit and discard changes, use the :q! keystrokes.

Step 2 `show rpl route-policy [name [detail] | states | brief]`

Example:

```
RP/0/# show rpl route-policy sample2
```

(Optional) Displays the configuration of a specific named route policy.

- Use the **detail** keyword to display all policies and sets that a policy uses.
- Use the **states** keyword to display all unused, inactive, and active states.
- Use the **brief** keyword to list the names of all extended community sets without their configurations.

Step 3 `show rpl prefix-set [name | states | brief]`

Example:

```
RP/0/# show rpl prefix-set prefixset1
```

(Optional) Displays the contents of a named prefix set.

- To display the contents of a named AS path set, community set, or extended community set, replace the **prefix-set** keyword with **as-path-set**, **community-set**, or **extcommunity-set**, respectively.

Simple Inbound Policy: Example

The following policy discards any route whose network layer reachability information (NLRI) specifies a prefix longer than /24, and any route whose NLRI specifies a destination in the address space reserved by RFC 1918. For all remaining routes, it sets the MED and local preference, and adds a community to the list in the route.

For routes whose community lists include any values in the range from 101:202 to 106:202 that have a 16-bit tag portion containing the value 202, the policy prepends autonomous system number 2 twice, and adds the community 2:666 to the list in the route. Of these routes, if the MED is either 666 or 225, then the policy sets the origin of the route to incomplete, and otherwise sets the origin to IGP.

For routes whose community lists do not include any of the values in the range from 101:202 to 106:202, the policy adds the community 2:999 to the list in the route.

```
prefix-set too-specific
 0.0.0.0/0 ge 25 le 32
end-set

prefix-set rfc1918
 10.0.0.0/8 le 32,
 172.16.0.0/12 le 32,
 192.168.0.0/16 le 32
end-set

route-policy inbound-tx
if destination in too-specific or destination in rfc1918 then
  drop
endif
set med 1000
set local-preference 90
set community (2:1001) additive
if community matches-any ([101..106]:202) then
  prepend as-path 2.30 2
  set community (2:666) additive
if med is 666 or med is 225 then
  set origin incomplete
else
  set origin igp
endif
else
  set community (2:999) additive
endif
end-policy

router bgp 2
 neighbor 10.0.1.2 address-family ipv4 unicast route-policy inbound-tx in
```

The following policy example shows how to build two inbound policies, in-100 and in-101, for two different peers. In building the specific policies for those peers, the policy reuses some common blocks of policy that may be common to multiple peers. It builds a few basic building blocks, the policies common-inbound, filter-bogons, and set-lpref-prepend.

The filter-bogons building block is a simple policy that filters all undesirable routes, such as those from the RFC 1918 address space. The policy set-lpref-prepend is a utility policy that can set the local preference and prepend the AS path according to parameterized values that are passed in. The common-inbound policy uses these filter-bogons building blocks to build a common block of inbound policy. The common-inbound policy is used as a building block in the construction of in-100 and in-101 along with the set-lpref-prepend building block.

```
prefix-set bogon
 10.0.0.0/8 ge 8 le 32,
 0.0.0.0,
 0.0.0.0/0 ge 27 le 32,
 192.168.0.0/16 ge 16 le 32
end-set
```

```
!  
route-policy in-100  
  apply common-inbound  
  if community matches-any ([100..120]:135) then  
    apply set-lpref-prepend (100,100,2)  
    set community (2:1234) additive  
  else  
    set local-preference 110  
  endif  
  if community matches-any ([100..666]:[100..999]) then  
    set med 444  
    set local-preference 200  
    set community (no-export) additive  
  endif  
end-policy  
!  
route-policy in-101  
  apply common-inbound  
  if community matches-any ([101..200]:201) then  
    apply set-lpref-prepend(100,101,2)  
    set community (2:1234) additive  
  else  
    set local-preference 125  
  endif  
end-policy  
!  
route-policy filter-bogons  
  if destination in bogon then  
  drop  
else  
pass  
  endif  
end-policy  
!  
route-policy common-inbound  
  apply filter-bogons  
  set origin igp  
  set community (2:333)  
end-policy  
!  
route-policy set-lpref-prepend($lpref,$as,$prependcnt)  
  set local-preference $lpref  
  prepend as-path $as $prependcnt  
end-policy
```

References for Routing Policy

To implement RPL, you need to understand the following concepts:

Routing Policy Language

This section contains the following information:

Routing Policy Language Overview

RPL was developed to support large-scale routing configurations. RPL has several fundamental capabilities that differ from those present in configurations oriented to traditional route maps, access lists, and prefix lists. The first of these capabilities is the ability to build policies in a modular form. Common blocks of policy can be defined and maintained independently. These common blocks of policy can then be applied from other blocks of policy to build complete policies. This capability reduces the amount of configuration information that needs to be maintained. In addition, these common blocks of policy can be parameterized. This parameterization allows for policies that share the same structure but differ in the specific values that are set or matched against to be maintained as independent blocks of policy. For example, three policies that are identical in every way except for the local preference value they set can be represented as one common parameterized policy that takes the varying local preference value as a parameter to the policy.

The policy language introduces the notion of sets. Sets are containers of similar data that can be used in route attribute matching and setting operations. Four set types exist: prefix-sets, community-sets, as-path-sets, and extcommunity-sets. These sets hold groupings of IPv4 or IPv6 prefixes, community values, AS path regular expressions, and extended community values, respectively. Sets are simply containers of data. Most sets also have an inline variant. An inline set allows for small enumerations of values to be used directly in a policy rather than having to refer to a named set. Prefix lists, community lists, and AS path lists must be maintained even when only one or two items are in the list. An inline set in RPL allows the user to place small sets of values directly in the policy body without having to refer to a named set.

Decision making, such as accept and deny, is explicitly controlled by the policy definitions themselves. RPL combines matching operators, which may use set data, with the traditional Boolean logic operators AND, OR, and NOT into complex conditional expressions. All matching operations return a true or false result. The execution of these conditional expressions and their associated actions can then be controlled by using simple *if then*, *elseif*, and *else* structures, which allow the evaluation paths through the policy to be fully specified by the user.

Routing Policy Language Structure

This section describes the basic structure of RPL.

Names

The policy language provides two kinds of persistent, namable objects: sets and policies. Definition of these objects is bracketed by beginning and ending command lines. For example, to define a policy named `test`, the configuration syntax would look similar to the following:

```
route-policy test
[ . . . policy statements . . . ]
end-policy
```

Legal names for policy objects can be any sequence of the upper- and lowercase alphabetic characters; the numerals 0 to 9; and the punctuation characters period, hyphen, and underscore. A name must begin with a letter or numeral.

Sets

In this context, the term set is used in its mathematical sense to mean an unordered collection of unique elements. The policy language provides sets as a container for groups of values for matching purposes. Sets

are used in conditional expressions. The elements of the set are separated by commas. Null (empty) sets are allowed.

In the following example:

```
prefix-set backup-routes
# currently no backup routes are defined
end-set
```

a condition such as:

```
if destination in backup-routes then
```

evaluates as FALSE for every route, because there is no match-condition in the prefix set that it satisfies.

You may want to perform comparisons against a small number of elements, such as two or three community values, for example. To allow for these comparisons, the user can enumerate these values directly. These enumerations are referred to as *inline sets*. Functionally, inline sets are equivalent to named sets, but allow for simple tests to be inline. Thus, comparisons do not require that a separate named set be maintained when only one or two elements are being compared. See the set types described in the following sections for the syntax. In general, the syntax for an inline set is a comma-separated list surrounded by parentheses, where element-entry is an entry of an item appropriate to the type of usage such as a prefix or a community value.

The following is an example using an inline community set:

```
route-policy sample-inline
if community matches-any ([10..15]:100) then
set local-preference 100
endif
end-policy
```

The following is an equivalent example using the named set test-communities:

```
community-set test-communities
10:100,
11:100,
12:100,
13:100,
14:100,
15:100
end-set

route-policy sample
if community matches-any test-communities then
set local-preference 100
endif
end-policy
```

Both of these policies are functionally equivalent, but the inline form does not require the configuration of the community set just to store the six values. You can choose the form appropriate to the configuration context. In the following sections, examples of both the named set version and the inline form are provided where appropriate.

as-path-set

An AS path set comprises operations for matching an AS path attribute. The only matching operation is a regular expression match.

Named Set Form

The named set form uses the **ios-regex** keyword to indicate the type of regular expression and requires single quotation marks around the regular expression.

The following is a sample definition of a named AS path set:

```
as-path-set aset1
ios-regex '_42$',
ios-regex '_127$'
end-set
```

This AS path set comprises two elements. When used in a matching operation, this AS path set matches any route whose AS path ends with either the autonomous system (AS) number 42 or 127.

To remove the named AS path set, use the **no as-path-set aset1** command-line interface (CLI) command.



Note Regular expression matching is CPU intensive. The policy performance can be substantially improved by either collapsing the regular expression patterns together to reduce the total number of regular expression invocations or by using equivalent native as-path match operations such as 'as-path neighbor-is', 'as-path originates-from' or 'as-path passes-through'.

Inline Set Form

The inline set form is a parenthesized list of comma-separated expressions, as follows:

```
(ios-regex '_42$', ios-regex '_127$')
```

This set matches the same AS paths as the previously named set, but does not require the extra effort of creating a named set separate from the policy that uses it.

community-set

A community-set holds community values for matching against the BGP community attribute. A community is a 32-bit quantity. Integer community values *must* be split in half and expressed as two unsigned decimal integers in the range from 0 to 65535, separated by a colon. Single 32-bit community values are not allowed. The following is the named set form:

Named Set Form

```
community-set cset1
12:34,
12:56,
12:78,
internet
```

```
end-set
```

Inline Set Form

```
(12:34, 12:56, 12:78)
($as:34, $as:$tag1, 12:78, internet)
```

The inline form of a community-set also supports parameterization. Each 16-bit portion of the community may be parameterized.

RPL provides symbolic names for the standard well-known community values: internet is 0:0, no-export is 65535:65281, no-advertise is 65535:65282, and local-as is 65535:65283.

RPL also provides a facility for using *wildcards* in community specifications. A wildcard is specified by inserting an asterisk (*) in place of one of the 16-bit portions of the community specification; the wildcard indicates that any value for that portion of the community matches. Thus, the following policy matches all communities in which the autonomous system part of the community is 123:

```
community-set cset3
  123:*
end-set
```

Every community set must contain at least one community value. Empty community sets are invalid and are rejected.

extcommunity-set

An extended community-set is analogous to a community-set except that it contains extended community values instead of regular community values. It also supports named forms and inline forms. There are three types of extended community sets: cost, soo, and rt.

As with community sets, the inline form supports parameterization within parameterized policies. Either portion of the extended community value can be parameterized.

Wildcards (*) and regular expressions are allowed for extended community set elements.

Every extended community-set must contain at least one extended community value. Empty extended community-sets are invalid and rejected.

The following are syntactic examples:

Named Form for Extcommunity-set RT

An rt set is an extcommunity set used to store BGP Route Target (RT) extended community type communities:

```
extcommunity-set rt a_rt_set
  1.2.3.4:666
  1234:666,
  1.2.3.4:777,
  4567:777
end-set
```

```
Inline Set Form for Extcommunity-set RT
```

```
(1.2.3.4:666, 1234:666, 1.2.3.4:777, 4567:777)
($ipaddr:666, 1234:$tag, 1.2.3.4:777, $tag2:777)
```

These options are supported under extended community set RT:

```
RP/0/ (config) #extcommunity-set rt rt_set
RP/0/ (config-ext) #?
#-remark      Remark beginning with '#'
*             Wildcard (any community or part thereof)
<1-4294967295> 32-bit decimal number
<1-65535>     16-bit decimal number
A.B.C.D/M:N   Extended community - IPv4 prefix format
A.B.C.D:N    Extended community - IPv4 format
ASN:N        Extended community - ASPLAIN format
X.Y:N        Extended community - ASDOT format
abort        Discard RPL definition and return to top level config
dfa-regex    DFA style regular expression
end-set      End of set definition
exit         Exit from this submode
ios-regex    Traditional IOS style regular expression
show        Show partial RPL configuration
```

Option	Description
#-remark	Remark beginning with '#'
*	Wildcard (any community or part thereof)
<1-4294967295>	32-bit decimal number
<1-65535>	16-bit decimal number
A.B.C.D/M:N	Extended community - IPv4 prefix format
A.B.C.D:N	Extended community - IPv4 format
ASN:N	Extended community - ASPLAIN format
X.Y:N	Extended community - ASDOT format
abort	Discard RPL definition and return to top level config
dfa-regex	DFA style regular expression
end-set	End of set definition
exit	Exit from this submode
ios-regex	Traditional IOS style regular expression
show	Show partial RPL configuration

Named Form for Extcommunity-set SoO

A soo set is an extcommunity set used to store BGP Site-of-Origin (SoO) extended community type communities:

```
extcommunity-set soo a_soo_set
1.1.1:100,
  100:200
```

```
end-set
```

These options are supported under extended community set Soo:

```
RP/0/ (config) #extcommunity-set soo soo_set
RP/0/ (config-ext) #?
#-remark      Remark beginning with '#'
*             Wildcard (any community or part thereof)
<1-4294967295> 32-bit decimal number
<1-65535>     16-bit decimal number
A.B.C.D/M:N   Extended community - IPv4 prefix format
A.B.C.D:N     Extended community - IPv4 format
ASN:N        Extended community - ASPLAIN format
X.Y:N        Extended community - ASDOT format
abort        Discard RPL definition and return to top level config
dfa-regex    DFA style regular expression
end-set      End of set definition
exit         Exit from this submode
ios-regex    Traditional IOS style regular expression
show        Show partial RPL configuration
```

Option	Description
#-remark	Remark beginning with '#'
*	Wildcard (any community or part thereof)
<1-4294967295>	32-bit decimal number
<1-65535>	16-bit decimal number
A.B.C.D/M:N	Extended community - IPv4 prefix format
A.B.C.D:N	Extended community - IPv4 format
ASN:N	Extended community - ASPLAIN format
X.Y:N	Extended community - ASDOT format
abort	Discard RPL definition and return to top level config
dfa-regex	DFA style regular expression
end-set	End of set definition
exit	Exit from this submode
ios-regex	Traditional IOS style regular expression
show	Show partial RPL configuration

prefix-set

A prefix-set holds IPv4 or IPv6 prefix match specifications, each of which has four parts: an address, a mask length, a minimum matching length, and a maximum matching length. The address is required, but the other three parts are optional. The address is a standard dotted-decimal IPv4 or colon-separated hexadecimal IPv6 address. The mask length, if present, is a nonnegative decimal integer in the range from 0 to 32 (0 to 128 for IPv6) following the address and separated from it by a slash. The optional minimum matching length follows the address and optional mask length and is expressed as the keyword **ge** (mnemonic for **greater than or equal to**), followed by a nonnegative decimal integer in the range from 0 to 32 (0 to 128 for IPv6). The optional

maximum matching length follows the rest and is expressed by the keyword **le** (mnemonic for **l**ess than or **e**qual to), followed by yet another nonnegative decimal integer in the range from 0 to 32 (0 to 128 for IPv6). A syntactic shortcut for specifying an exact length for prefixes to match is the **eq** keyword (mnemonic for **e**qual to).

If a prefix match specification has no mask length, then the default mask length is 32 for IPv4 and 128 for IPv6. The default minimum matching length is the mask length. If a minimum matching length is specified, then the default maximum matching length is 32 for IPv4 and 128 for IPv6. Otherwise, if neither minimum nor maximum is specified, the default maximum is the mask length.

The prefix-set itself is a comma-separated list of prefix match specifications. The following are examples:

```
prefix-set legal-ipv4-prefix-examples
  10.0.1.1,
  10.0.2.0/24,
  10.0.3.0/24 ge 28,
  10.0.4.0/24 le 28,
  10.0.5.0/24 ge 26 le 30,
  10.0.6.0/24 eq 28,
  10.0.7.2/32 ge 16 le 24,
  10.0.8.0/26 ge 8 le 16
end-set

prefix-set legal-ipv6-prefix-examples
  2001:0:0:1::/64,
  2001:0:0:2::/64 ge 96,
  2001:0:0:2::/64 ge 96 le 100,
  2001:0:0:2::/64 eq 100
end-set
```

The first element of the prefix-set matches only one possible value, 10.0.1.1/32 or the host address 10.0.1.1. The second element matches only one possible value, 10.0.2.0/24. The third element matches a range of prefix values, from 10.0.3.0/28 to 10.0.3.255/32. The fourth element matches a range of values, from 10.0.4.0/24 to 10.0.4.240/28. The fifth element matches prefixes in the range from 10.0.5.0/26 to 10.0.5.252/30. The sixth element matches any prefix of length 28 in the range from 10.0.6.0/28 through 10.0.6.240/28. The seventh element matches any prefix of length 32 in the range 10.0.[0..255].2/32 (from 10.0.0.2/32 to 10.0.255.2). The eighth element matches any prefix of length 26 in the range 10.[0..255].8.0/26 (from 10.0.8.0/26 to 10.255.8.0/26).

The following prefix-set consists entirely of invalid prefix match specifications:

```
prefix-set ILLEGAL-PREFIX-EXAMPLES
  10.1.1.1 ge 16,
  10.1.2.1 le 16,
  10.1.3.0/24 le 23,
  10.1.4.0/24 ge 33,
  10.1.5.0/25 ge 29 le 28
end-set
```

Neither the minimum length nor maximum length is valid without a mask length. For IPv4, the minimum length must be less than 32, the maximum length of an IPv4 prefix. For IPv6, the minimum length must be less than 128, the maximum length of an IPv6 prefix. The maximum length must be equal to or greater than the minimum length.

ACL Support in RPL Prefix Sets

Access Control List (ACL) type prefix set entries holds IPv4 or IPv6 prefix match specifications, each of which has an address and a wildcard mask. The address and wildcard mask is a standard dotted-decimal IPv4 or colon-separated hexadecimal IPv6 address. The set of bits to be matched are provided in the form of wildcard also called as inverted mask in which a binary 0 means a mandatory match and binary 1 means a do not match condition. The prefix set allows to specify contiguous and non-contiguous set of bits that should be matched in any route.

rd-set

An rd-set is used to create a set with route distinguisher (RD) elements. An RD set is a 64-bit value prepended to an IPv4 address to create a globally unique Border Gateway Protocol (BGP) VPN IPv4 address.

You can define RD values with the following commands:

- *a.b.c.d:m*:*—BGP VPN RD in IPv4 format with a wildcard character. For example, 10.0.0.2:255.255.0.0:*
- *a.b.c.d/m:n*—BGP VPN RD in IPv4 format with a mask. For example, 10.0.0.2:255.255.0.0:666.
- *a.b.c.d:***—BGP VPN RD in IPv4 format with a wildcard character. For example, 10.0.0.2:255.255.0.0.
- *a.b.c.d:n*—BGP VPN RD in IPv4 format. For example, 10.0.0.2:666.
- *asn:**—BGP VPN RD in ASN format with a wildcard character. For example, 10002:255.255.0.0.
- *asn:n*—BGP VPN RD in ASN format. For example, 10002:666.

The following is an example of an rd-set:

```
rd-set rdset1
  10.0.0.0/8:*,
  10.0.0.0/8:777,
  10.0.0.0:*,
  10.0.0.0:777,
  65000:*,
  65000:777
end-set
```

Routing Policy Language Components

Four main components in the routing policy language are involved in defining, modifying, and using policies: the configuration front end, policy repository, execution engine, and policy clients themselves.

The configuration front end (CLI) is the mechanism to define and modify policies. This configuration is then stored on the router using the normal storage means and can be displayed using the normal configuration **show** commands.

The second component of the policy infrastructure, the policy repository, has several responsibilities. First, it compiles the user-entered configuration into a form that the execution engine can understand. Second, it performs much of the verification of policies; and it ensures that defined policies can actually be executed properly. Third, it tracks which attach points are using which policies so that when policies are modified the appropriate clients are properly updated with the new policies relevant to them.

The third component is the execution engine. This component is the piece that actually runs policies as the clients request. The process can be thought of as receiving a route from one of the policy clients and then executing the actual policy against the specific route data.

The fourth component is the policy clients (the routing protocols). This component calls the execution engine at the appropriate times to have a given policy be applied to a given route, and then perform some number of actions. These actions may include deleting the route if policy indicated that it should be dropped, passing along the route to the protocol decision tree as a candidate for the best route, or advertising a policy modified route to a neighbor or peer as appropriate.

Routing Policy Language Usage

This section provides basic routing policy language usage examples.

Pass Policy

The following example shows how the policy accepts all presented routes without modifying the routes.

```
route-policy quickstart-pass
pass
end-policy
```

Drop Everything Policy

The following example shows how the policy explicitly rejects all routes presented to it. This type of policy is used to ignore everything coming from a specific peer.

```
route-policy quickstart-drop
drop
end-policy
```

Ignore Routes with Specific AS Numbers in the Path

The following example shows the policy definition in three parts. First, the **as-path-set** command defines three regular expressions to match against an AS path. Second, the **route-policy** command applies the AS path set to a route. If the AS path attribute of the route matches the regular expression defined with the **as-path-set** command, the protocol refuses the route. Third, the route policy is attached to BGP neighbor 10.0.1.2. BGP consults the policy named `ignore_path_as` on routes received (imported) from neighbor 10.0.1.2.

```
as-path-set ignore_path
ios-regex '_11_',
ios-regex '_22_',
ios-regex '_33_'
end-set

route-policy ignore_path_as
if as-path in ignore_path then
drop
else
pass
endif
end-policy

router bgp 2
neighbor 10.0.1.2 address-family ipv4 unicast policy ignore_path_as in
```


Set Community Based on MED

The following example shows how the policy tests the MED of a route and modifies the community attribute of the route based on the value of the MED. If the MED value is 127, the policy adds the community 123:456 to the route. If the MED value is 63, the policy adds the value 123:789 to the community attribute of the route. Otherwise, the policy removes the community 123:123 from the route. In any case, the policy instructs the protocol to accept the route.

```
route-policy quickstart-med
if med eq 127 then
set community (123:456) additive
elseif med eq 63 then
set community (123:789) additive
else
delete community in (123:123)
endif
pass
end-policy
```

Set Local Preference Based on Community

The following example shows how the community-set named quickstart-communities defines community values. The route policy named quickstart-localpref tests a route for the presence of the communities specified in the quickstart-communities community set. If any of the community values are present in the route, the route policy sets the local preference attribute of the route to 31. In any case, the policy instructs the protocol to accept the route.

```
community-set quickstart-communities
987:654,
987:543,
987:321,
987:210
end-set

route-policy quickstart-localpref
if community matches-any quickstart-communities then
set local-preference 31
endif
pass
end-policy
```

Persistent Remarks

The following example shows how comments are placed in the policy to clarify the meaning of the entries in the set and the statements in the policy. The remarks are persistent, meaning they remain attached to the policy. For example, remarks are displayed in the output of the **show running-config** command. Adding remarks to the policy makes the policy easier to understand, modify at a later date, and troubleshoot if an unexpected behavior occurs.

```
prefix-set rfc1918
# These are the networks defined as private in RFC1918 (including
# all subnets thereof)
10.0.0.0/8 ge 8,
172.16.0.0/12 ge 12,
192.168.0.0/16 ge 16
end-set
```

```

route-policy quickstart-remarks
# Handle routes to RFC1918 networks
if destination in rfc1918 then
# Set the community such that we do not export the route
set community (no-export) additive

endif
end-policy

```

Policy Definitions

Policy definitions create named sequences of policy statements. A policy definition consists of the CLI **route-policy** keyword followed by a name, a sequence of policy statements, and the **end-policy** keyword. For example, the following policy drops any route it encounters:

```

route-policy drop-everything
drop
end-policy

```

The name serves as a handle for binding the policy to protocols. To remove a policy definition, issue the **no route-policy name** command.

Policies may also refer to other policies such that common blocks of policy can be reused. This reference to other policies is accomplished by using the **apply** statement, as shown in the following example:

```

route-policy check-as-1234
if as-path passes-through '1234.5' then
apply drop-everything
else
pass
endif
end-policy

```

The **apply** statement indicates that the policy drop-everything should be executed if the route under consideration passed through autonomous system 1234.5 before it is received. If a route that has autonomous system 1234.5 in its AS path is received, the route is dropped; otherwise, the route is accepted without modification. This policy is an example of a hierarchical policy. Thus, the semantics of the **apply** statement are just as if the applied policy were cut and pasted into the applying policy:

```

route-policy check-as-1234-prime
if as-path passes-through '1234.5' then
drop
else
pass
endif
end-policy

```

You may have as many levels of hierarchy as desired. However, many levels may be difficult to maintain and understand.

Parameterization

In addition to supporting reuse of policies using the **apply** statement, policies can be defined that allow for parameterization of some of the attributes. The following example shows how to define a parameterized policy named `param-example`. In this case, the policy takes one parameter, `$mytag`. Parameters always begin with a dollar sign and consist otherwise of any alphanumeric characters. Parameters can be substituted into any attribute that takes a parameter.

In the following example, a 16-bit community tag is used as a parameter:

```
route-policy param-example ($mytag)
set community (1234:$mytag) additive
end-policy
```

This parameterized policy can then be reused with different parameterization, as shown in the following example. In this manner, policies that share a common structure but use different values in some of their individual statements can be modularized. For details on which attributes can be parameterized, see the individual attribute sections.

```
route-policy origin-10
if as-path originates-from '10.5' then
apply param-example(10.5)
else
pass
endif
end-policy

route-policy origin-20
if as-path originates-from '20.5' then
apply param-example(20.5)
else
pass
endif
end-policy
```

The parameterized policy `param-example` provides a policy definition that is expanded with the values provided as the parameters in the `apply` statement. Note that the policy hierarchy is always maintained. Thus, if the definition of `param-example` changes, then the behavior of `origin_10` and `origin_20` changes to match.

The effect of the `origin-10` policy is that it adds the community `1234:10` to all routes that pass through this policy and have an AS path indicating the route originated from autonomous system 10. The `origin-20` policy is similar except that it adds to community `1234:20` for routes originating from autonomous system 20.

Parameterization at Attach Points

In addition to supporting parameterization using the `apply` statement, policies can also be defined that allow for parameterization the attributes at attach points. Parameterization is supported at all attach points.

In the following example, we define a parameterized policy "param-example". In this example, the policy takes two parameters "\$mymed" and "\$prefixset". Parameters always begin with a dollar sign, and consist otherwise of any alphanumeric characters. Parameters can be substituted into any attribute that takes a parameter. In this example we are passing a MED value and prefix set name as parameters.

```
route-policy param-example ($mymed, $prefixset)
  if destination in $prefixset then
    set med $mymed
  endif
end-policy
```

This parameterized policy can then be reused with different parameterizations as shown in the example below. In this manner, policies that share a common structure but use different values in some of their individual statements can be modularized. For details on which attributes can be parameterized, see the individual attributes for each protocol.

```
router bgp 2
  neighbor 10.1.1.1
    remote-as 3
    address-family ipv4 unicast
      route-policy param-example(10, prefix_set1)
      route-policy param-example(20, prefix_set2)
```

The parameterized policy `param-example` provides a policy definition that is expanded with the values provided as the parameters in the `neighbor route-policy in and out` statement.

Global Parameterization

RPL supports the definition of systemwide global parameters that can be used inside policy definition. Global parameters can be configured as follows:

```
Policy-global
  glbpathtype 'ebgp'
  glbtag '100'
end-global
```

The global parameter values can be used directly inside a policy definition similar to the local parameters of parameterized policy. In the following example, the *globalparam* argument, which makes use of the global parameters `glbpathtype` and `glbtag`, is defined for a nonparameterized policy.

```
route-policy globalparam
  if path-type is $glbpathtype then
    set tag $glbtag
  endif
end-policy
```

When a parameterized policy has a parameter name “collision” with a global parameter name, parameters local to policy definition take precedence, effectively masking off global parameters. In addition, a validation mechanism is in place to prevent the deletion of a particular global parameter if it is referred by any policy.

Semantics of Policy Application

This section discusses how routing policies are evaluated and applied. The following concepts are discussed:

Boolean Operator Precedence

Boolean expressions are evaluated in order of operator precedence, from left to right. The highest precedence operator is NOT, followed by AND, and then OR. The following expression:

```
med eq 10 and not destination in (10.1.3.0/24) or community matches-any ([10..25]:35)
```

if fully parenthesized to display the order of evaluation, would look like this:

```
(med eq 10 and (not destination in (10.1.3.0/24))) or community matches-any ([10..25]:35)
```

The inner NOT applies only to the destination test; the AND combines the result of the NOT expression with the Multi Exit Discriminator (MED) test; and the OR combines that result with the community test. If the order of operations are rearranged:

```
not med eq 10 and destination in (10.1.3.0/24) or community matches-any ([10..25]:35)
```

then the expression, fully parenthesized, would look like the following:

```
((not med eq 10) and destination in (10.1.3.0/24)) or community matches-any ([10..25]:35)
```

Multiple Modifications of Same Attribute

When a policy replaces the value of an attribute multiple times, the last assignment wins because all actions are executed. Because the MED attribute in BGP is one unique value, the last value to which it gets set to wins. Therefore, the following policy results in a route with a MED value of 12:

```
set med 9
set med 10
set med 11
set med 12
```

This example is trivial, but the feature is not. It is possible to write a policy that effectively changes the value for an attribute. For example:

```
set med 8
if community matches-any cs1 then
set local-preference 122
if community matches-any cs2 then
set med 12
endif
endif
```

The result is a route with a MED of 8, unless the community list of the route matches both cs1 and cs2, in which case the result is a route with a MED of 12.

In the case in which the attribute being modified can contain only one value, it is easy to think of this case as the last statement wins. However, a few attributes can contain multiple values and the result of multiple actions

on the attribute is cumulative rather than as a replacement. The first of these cases is the use of the **additive** keyword on community and extended community evaluation. Consider a policy of the form:

```
route-policy community-add
set community (10:23)
set community (10:24) additive
set community (10:25) additive
end-policy
```

This policy sets the community string on the route to contain all three community values: 10:23, 10:24, and 10:25.

The second of these cases is AS path prepending. Consider a policy of the form:

```
route-policy prepend-example
prepend as-path 2.5 3
prepend as-path 666.5 2
end-policy
```

This policy prepends 666.5 666.5 2.5 2.5 2.5 to the AS path. This prepending is a result of all actions being taken and to the AS path being an attribute that contains an array of values rather than a simple scalar value.

When Attributes Are Modified

A policy does not modify route attribute values until all tests have been completed. In other words, comparison operators always run on the initial data in the route. Intermediate modifications of the route attributes do not have a cascading effect on the evaluation of the policy. Take the following example:

```
ifmed eq 12 then
set med 42
if med eq 42 then
drop
endif
endif
```

This policy never executes the drop statement because the second test (med eq 42) sees the original, unmodified value of the MED in the route. Because the MED has to be 12 to get to the second test, the second test always returns false.

Default Drop Disposition

All route policies have a default action to drop the route under evaluation unless the route has been modified by a policy action or explicitly passed. Applied (nested) policies implement this disposition as though the applied policy were pasted into the point where it is applied.

Consider a policy to allow all routes in the 10 network and set their local preference to 200 while dropping all other routes. You might write the policy as follows:

```
route-policy two
if destination in (10.0.0.0/8 ge 8 le 32) then
set local-preference 200
endif
```

```
end-policy

route-policy one
apply two
end-policy
```

It may appear that policy one drops all routes because it neither contains an explicit **pass** statement nor modifies a route attribute. However, the applied policy does set an attribute for some routes and this disposition is passed along to policy one. The result is that policy one passes routes with destinations in network 10, and drops all others.

Control Flow

Policy statements are processed sequentially in the order in which they appear in the configuration. Policies that hierarchically reference other policy blocks are processed as if the referenced policy blocks had been directly substituted inline. For example, if the following policies are defined:

```
route-policy one
set weight 100
end-policy

route-policy two
set med 200
end-policy

route-policy three
apply two
set community (2:666) additive
end-policy

route-policy four
apply one
apply three
pass
end-policy
```

Policy four could be rewritten in an equivalent way as follows:

```
route-policy four-equivalent
set weight 100
set med 200
set community (2:666) additive
pass
end-policy
```



Note The **pass** statement is not required and can be removed to represent the equivalent policy in another way.

Policy Verification

Several different types of verification occur when policies are being defined and used.

Range Checking

As policies are being defined, some simple verifications, such as range checking of values, is done. For example, the MED that is being set is checked to verify that it is in a proper range for the MED attribute. However, this range checking cannot cover parameter specifications because they may not have defined values yet. These parameter specifications are verified when a policy is attached to an attach point. The policy repository also verifies that there are no recursive definitions of policy, and that parameter numbers are correct. At attach time, all policies must be well formed. All sets and policies that they reference must be defined and have valid values. Likewise, any parameter values must also be in the proper ranges.

Incomplete Policy and Set References

As long as a given policy is not attached at an attach point, the policy is allowed to refer to nonexistent sets and policies, which allows for freedom of workflow. You can build configurations that reference sets or policy blocks that are not yet defined, and then can later fill in those undefined policies and sets, thereby achieving much greater flexibility in policy definition. Every piece of policy you want to reference while defining a policy need not exist in the configuration. Thus, a user can define a policy sample that references the policy bar using an **apply** statement even if the policy bar does not exist. Similarly, a user can enter a policy statement that refers to a nonexistent set.

However, the existence of all referenced policies and sets is enforced when a policy is attached. If you attempt to attach the policy sample with the reference to an undefined policy bar at an inbound BGP policy using the **neighbor 1.2.3.4 address-family ipv4 unicast policy sample in** command, the configuration attempt is rejected because the policy bar does not exist.

Likewise, you cannot remove a route policy or set that is currently in use at an attach point because this removal would result in an undefined reference. An attempt to remove a route policy or set that is currently in use results in an error message to the user.

A condition exists that is referred to as a null policy in which the policy bar exists but has no statements, actions, or dispositions in it. In other words, the policy bar does exist as follows:

```
route-policy bar
end-policy
```

This is a valid policy block. It effectively forces all routes to be dropped because it is a policy block that never modifies a route, nor does it include the pass statement. Thus, the default action of drop for the policy block is followed.

Aggregation

The aggregation attach point generates an aggregate route to be advertised based on the conditional presence of subcomponents of that aggregate. Policies attached at this attach point are also able to set any of the valid BGP attributes on the aggregated routes. For example, the policy could set a community value or a MED on the aggregate that is generated. The specified aggregate is generated if any routes evaluated by the named policy pass the policy. More specifics of the aggregate are filtered using the **suppress-route** keyword. Any actions taken to set attributes in the route affect attributes on the aggregate.

In the policy language, the configuration is controlled by which routes pass the policy. The suppress map was used to selectively filter or suppress specific components of the aggregate when the summary-only flag is not set. In other words, when the aggregate and more specific components are being sent, some of the more specific components can be filtered using a suppress map. In the policy language, this is controlled by selecting the route and setting the suppress flag. The attribute-map allowed the user to set specific attributes on the

aggregated route. In the policy language, setting attributes on the aggregated route is controlled by normal action operations.

In the following example, the aggregate address 10.0.0.0/8 is generated if there are any component routes in the range 10.0.0.0/8 ge 8 le 25 except for 10.2.0.0/24. Because summary-only is not set, all components of the aggregate are advertised. However, the specific component 10.1.0.0 are suppressed.

```
route-policy sample
  if destination in (10.0.0.0/8 ge 8 le 25) then
    set community (10:33)
  endif
  if destination in (10.2.0.0/24) then
    drop
  endif
  if destination in (10.1.0.0/24) then
    suppress-route
  endif
end-policy

router bgp 2
address-family ipv4
  aggregate-address 10.0.0.0/8 route-policy sample
  .
  .
  .
```

The effect of aggregation policy on the attributes of the aggregate is cumulative. Every time an aggregation policy matches a more specific route, the set operations in the policy may modify the aggregate. The aggregate in the following example has a MED value that varies according to the number of more specific routes that comprise the aggregate.

```
route-policy bumping-aggregation
  set med +5
end-policy
```

If there are three matching more specific routes, the MED of the aggregate is the default plus 15; if there are seventeen more specific routes, the MED of the aggregate is the default plus 85.

The order that the aggregation policy is applied to prefix paths is deterministic but unspecified. That is, a given set of routes always appears in the same order, but there is no way to predict the order.

A drop in aggregation policy does not prevent generation of an aggregate, but it does prevent the current more specific route from contributing to the aggregate. If another more specific route gives the route a pass, the aggregate is generated. Only one more specific pass is required to generate an aggregate.

Policy Statements

Four types of policy statements exist: remark, disposition (drop and pass), action (set), and if (comparator).

Remark

A remark is text attached to policy configuration but otherwise ignored by the policy language parser. Remarks are useful for documenting parts of a policy. The syntax for a remark is text that has each line prepended with a pound sign (#):

```
# This is a simple one-line remark.

# This
# is a remark
# comprising multiple
# lines.
```

In general, remarks are used between complete statements or elements of a set. Remarks are not supported in the middle of statements or within an inline set definition.

Unlike traditional !-comments in the CLI, RPL remarks persist through reboots and when configurations are saved to disk or a TFTP server and then loaded back onto the router.

Disposition

If a policy modifies a route, by default the policy accepts the route. RPL provides a statement to force the opposite—the **drop** statement. If a policy matches a route and executes a drop, the policy does not accept the route. If a policy does not modify the route, by default the route is dropped. To prevent the route from being dropped, the **pass** statement is used.

The **drop** statement indicates that the action to take is to discard the route. When a route is dropped, no further execution of policy occurs. For example, if after executing the first two statements of a policy the **drop** statement is encountered, the policy stops and the route is discarded.



Note All policies have a default **drop** action at the end of execution.

The **pass** statement allows a policy to continue executing even though the route has not been modified. When a policy has finished executing, any route that has been modified in the policy or any route that has received a pass disposition in the policy, successfully passes the policy and completes the execution. If route policy B_rp is applied within route policy A_rp, execution continues from policy A_rp to policy B_rp and back to policy A_rp provided prefix is not dropped by policy B_rp.

```
route-policy A_rp
  set community (10:10)
  apply B_rp
end-policy
!

route-policy B_rp
  if destination in (121.23.0.0/16 le 32, 155.12.0.0/16 le 32) then
    set community (121:155) additive
  endif
end-policy
!
```

By default, a route is **dropped** at the end of policy processing unless either the policy **modifies** a route attribute or it passes the route by means of an explicit **pass** statement. For example, if route-policy B is applied within route-policy A, then execution continues from policy A to policy B and back to policy A, provided the prefix is not dropped by policy B.

```
route-policy A
  if as-path neighbor-is '123' then
```

```
    apply B
    policy statement N
end-policy
```

Whereas the following policies pass all routes that they evaluate.

```
route-policy PASS-ALL
pass
end-policy
```

```
route-policy SET-LPREF
set local-preference 200
end-policy
```

In addition to being implicitly dropped, a route may be dropped by an **explicit drop** statement. **Drop** statements cause a route to be dropped immediately so that no further policy processing is done. Note also that a **drop** statement overrides any previously processed **pass** statements or attribute modifications. For example, the following policy drops all routes. The first **pass** statement is executed, but is then immediately overridden by the **drop** statement. The second **pass** statement never gets executed.

```
route-policy DROP-EXAMPLE
pass
drop
pass
end-policy
```

When one policy applies another, it is as if the applied policy were copied into the right place in the applying policy, and then the same drop-and-pass semantics are put into effect. For example, policies ONE and TWO are equivalent to policy ONE-PRIME:

```
route-policy ONE
  apply two
  if as-path neighbor-is '123' then
  pass
  endif
end-policy

route-policy TWO
  if destination in (10.0.0.0/16 le 32) then
  drop
  endif
end-policy

route-policy ONE-PRIME
  if destination in (10.0.0.0/16 le 32) then
  drop
  endif
  if as-path neighbor-is '123' then
  pass
  endif
end-policy
```

Because the effect of an **explicit drop** statement is immediate, routes in 10.0.0.0/16 le 32 are dropped without any further policy processing. Other routes are then considered to see if they were advertised by autonomous system 123. If they were advertised, they are passed; otherwise, they are implicitly dropped at the end of all policy processing.

The **done** statement indicates that the action to take is to stop executing the policy and accept the route. When encountering a **done** statement, the route is passed and no further policy statements are executed. All modifications made to the route prior to the **done** statement are still valid.

Action

An action is a sequence of primitive operations that modify a route. Most actions, but not all, are distinguished by the **set** keyword. In a route policy, actions can be grouped together. For example, the following is a route policy comprising three actions:

```
route-policy actions
set med 217
set community (12:34) additive
delete community in (12:56)
end-policy
```

If

In its simplest form, an **if** statement uses a conditional expression to decide which actions or dispositions should be taken for the given route. For example:

```
if as-path in as-path-set-1 then
drop
endif
```

The example indicates that any routes whose AS path is in the set as-path-set-1 are dropped. The contents of the **then** clause may be an arbitrary sequence of policy statements.

The following example contains two action statements:

```
if origin is igp then
set med 42
prepend as-path 73.5 5
endif
```

The CLI provides support for the **exit** command as an alternative to the **endif** command.

The **if** statement also permits an **else** clause, which is executed if the if condition is false:

```
if med eq 8 then
set community (12:34) additive
else
set community (12:56) additive
endif
```

The policy language also provides syntax, using the **elseif** keyword, to string together a sequence of tests:

```
if med eq 150 then
set local-preference 10
elseif med eq 200 then
set local-preference 60
elseif med eq 250 then
set local-preference 110
else
set local-preference 0
endif
```

The statements within an **if** statement may themselves be **if** statements, as shown in the following example:

```
if community matches-any (12:34,56:78) then
if med eq 150 then
drop
endif
set local-preference 100
endif
```

This policy example sets the value of the local preference attribute to 100 on any route that has a community value of 12:34 or 56:78 associated with it. However, if any of these routes has a MED value of 150, then these routes with either the community value of 12:34 or 56:78 and a MED of 150 are dropped.

Boolean Conditions

In the previous section describing the **if** statement, all of the examples use simple Boolean conditions that evaluate to either true or false. RPL also provides a way to build compound conditions from simple conditions by means of Boolean operators.

Three Boolean operators exist: negation (**not**), conjunction (**and**), and disjunction (**or**). In the policy language, negation has the highest precedence, followed by conjunction, and then by disjunction. Parentheses may be used to group compound conditions to override precedence or to improve readability.

The following simple condition:

```
med eq 42
```

is true only if the value of the MED in the route is 42, otherwise it is false.

A simple condition may also be negated using the **not** operator:

```
not next-hop in (10.0.2.2)
```

Any Boolean condition enclosed in parentheses is itself a Boolean condition:

```
(destination in prefix-list-1)
```

A compound condition takes either of two forms. It can be a simple expression followed by the **and** operator, itself followed by a simple condition:

```
med eq 42 and next-hop in (10.0.2.2)
```

A compound condition may also be a simpler expression followed by the **or** operator and then another simple condition:

```
origin is igp or origin is incomplete
```

An entire compound condition may be enclosed in parentheses:

```
(med eq 42 and next-hop in (10.0.2.2))
```

The parentheses may serve to make the grouping of subconditions more readable, or they may force the evaluation of a subcondition as a unit.

In the following example, the highest-precedence **not** operator applies only to the destination test, the **and** operator combines the result of the **not** expression with the community test, and the **or** operator combines that result with the MED test.

```
med eq 10 or not destination in (10.1.3.0/24) and community matches-any ([12..34]:[56..78])
```

With a set of parentheses to express the precedence, the result is the following:

```
med eq 10 or ((not destination in (10.1.3.0/24)) and community matches-any ([12..34]:[56..78]))
```

The following is another example of a complex expression:

```
(origin is igp or origin is incomplete or not med eq 42) and next-hop in (10.0.2.2)
```

The left conjunction is a compound condition enclosed in parentheses. The first simple condition of the inner compound condition tests the value of the origin attribute; if it is Interior Gateway Protocol (IGP), then the inner compound condition is true. Otherwise, the evaluation moves on to test the value of the origin attribute again, and if it is incomplete, then the inner compound condition is true. Otherwise, the evaluation moves to check the next component condition, which is a negation of a simple condition.

apply

As discussed in the sections on policy definitions and parameterization of policies, the **apply** command executes another policy (either parameterized or unparameterized) from within another policy, which allows for the reuse of common blocks of policy. When combined with the ability to parameterize common blocks of policy, the **apply** command becomes a powerful tool for reducing repetitive configuration.

Attach Points

Policies do not become useful until they are applied to routes, and for policies to be applied to routes they need to be made known to routing protocols. In BGP, for example, there are several situations where policies can be used, the most common of these is defining import and export policy. The policy attach point is the point in which an association is formed between a specific protocol entity, in this case a BGP neighbor, and a specific named policy. It is important to note that a verification step happens at this point. Each time a policy is attached, the given policy and any policies it may apply are checked to ensure that the policy can be validly used at that attach point. For example, if a user defines a policy that sets the IS-IS level attribute and then attempts to attach this policy as an inbound BGP policy, the attempt would be rejected because BGP routes do not carry IS-IS attributes. Likewise, when policies are modified that are in use, the attempt to modify the policy is verified against all current uses of the policy to ensure that the modification is compatible with the current uses.

Each protocol has a distinct definition of the set of attributes (commands) that compose a route. For example, BGP routes may have a community attribute, which is undefined in OSPF. Routes in IS-IS have a level attribute, which is unknown to BGP. Routes carried internally in the RIB may have a tag attribute.

When a policy is attached to a protocol, the protocol checks the policy to ensure the policy operates using route attributes known to the protocol. If the protocol uses unknown attributes, then the protocol rejects the attachment. For example, OSPF rejects attachment of a policy that tests the values of BGP communities.

The situation is made more complex by the fact that each protocol has access to at least two distinct route types. In addition to native protocol routes, for example BGP or IS-IS, some protocol policy attach points operate on RIB routes, which is the common central representation. Using BGP as an example, the protocol provides an attach point to apply policy to routes redistributed from the RIB to BGP. An attach point dealing with two different kinds of routes permits a mix of operations: RIB attribute operations for matching and BGP attribute operations for setting.



Note The protocol configuration rejects attempts to attach policies that perform unsupported operations.

The following sections describe the protocol attach points, including information on the attributes (commands) and operations that are valid for each attach point.

BGP Policy Attach Points

This section describes each of the BGP policy attach points and provides a summary of the BGP attributes and operators.

Additional-Path

The additional-path attach point provides increased control based on various attribute match operations. This attach point is used to decide whether a route-policy should be used to select additional-paths for a BGP speaker to be able to send multiple paths for the prefix.

The add path enables BGP prefix independent convergence (PIC) at the edge routers.

This example shows how to set a route-policy "add-path-policy" to be used for enabling selection of additional paths:

```
router bgp 100
  address-family ipv4 unicast
    additional-paths selection route-policy add-path-policy
```

Dampening

The dampening attach point controls the default route-dampening behavior within BGP. Unless overridden by a more specific policy on the associate peer, all routes in BGP apply the associated policy to set their dampening attributes.

The following policy sets dampening values for BGP IPv4 unicast routes. Those routes that are more specific than a /25 take longer to recover after they are dampened than the routes that are less specific than /25.



- Note** When the dampening policy runs for a route, then the last "set dampening" statement that is encountered, takes effect.
- If a "drop" statement is encountered, then the route is not dampened; even if the "set dampening" statement is encountered.
 - If a "pass" or "done" statement is encountered but not the "set dampening" statement, then the route is dampened using the default dampening parameters.

For example:

- When policy1 applies another policy that is called policy2 and if a "pass" statement is encountered in policy2, then policy2 exits and continues to execute policy1.
- If a "done" statement is encountered in policy2, then both policy1 and policy2 exits immediately.

```
route-policy sample_damp
  if destination in (0.0.0.0/0 ge 25) then
    set dampening halflife 30 others default
  else
    set dampening halflife 20 others default
  endif
end-policy

router bgp 2
  address-family ipv4 unicast
    bgp dampening route-policy sample_damp
  .
  .
  .
```

Default Originate

The default originate attach point allows the default route (0.0.0.0/0) to be conditionally generated and advertised to a peer, based on the presence of other routes. It accomplishes this configuration by evaluating the associated policy against routes in the Routing Information Base (RIB). If any routes pass the policy, the default route is generated and sent to the relevant peer.

The following policy generates and sends a default-route to the BGP neighbor 10.0.0.1 if any routes that match 10.0.0.0/8 ge 8 le 32 are present in the RIB.

```
route-policy sample-originate
  if rib-has-route in (10.0.0.0/8 ge 8 le 32) then
    pass
  endif
end-policy
```



```
router bgp 2
  neighbor 10.0.0.1
    remote-as 3
    address-family ipv4 unicast
    default-originate policy sample-originate
  .
  .
  .
```

Neighbor Export

The neighbor export attach point selects the BGP routes to send to a given peer or group of peers. The routes are selected by running the set of possible BGP routes through the associated policy. Any routes that pass the policy are then sent as updates to the peer or group of peers. The routes that are sent may have had their BGP attributes altered by the policy that has been applied.

The following policy sends all BGP routes to neighbor 10.0.0.5. Routes that are tagged with any community in the range 2:100 to 2:200 are sent with a MED of 100 and a community of 2:666. The rest of the routes are sent with a MED of 200 and a community of 2:200.

```
route-policy sample-export
  if community matches-any (2:[100-200]) then
    set med 100
    set community (2:666)
  else
    set med 200
    set community (2:200)
  endif
end-policy

router bgp 2
  neighbor 10.0.0.5
    remote-as 3
    address-family ipv4 unicast
    route-policy sample-export out
  .
  .
  .
```

Neighbor Import

The neighbor import attach point controls the reception of routes from a specific peer. All routes that are received by a peer are run through the attached policy. Any routes that pass the attached policy are passed to the BGP Routing Information Base (BRIB) as possible candidates for selection as best path routes.

When a BGP import policy is modified, it is necessary to rerun all the routes that have been received from that peer against the new policy. The modified policy may now discard routes that were previously allowed through, allow through previously discarded routes, or change the way the routes are modified. A new configuration option in BGP (**bgp auto-policy-soft-reset**) that allows this modification to happen automatically in cases for which either soft reconfiguration is configured or the BGP route-refresh capability has been negotiated.

The following example shows how to receive routes from neighbor 10.0.0.1. Any routes received with the community 3:100 have their local preference set to 100 and their community tag set to 2:666. All other routes received from this peer have their local preference set to 200 and their community tag set to 2:200.

```

route-policy sample_import
  if community matches-any (3:100) then
    set local-preference 100
    set community (2:666)
  else
    set local-preference 200
    set community (2:200)
  endif
end-policy

router bgp 2
  neighbor 10.0.0.1
  remote-as 3
  address-family ipv4 unicast
    route-policy sample_import in
  .
  .
  .

```

Network

The network attach point controls the injection of routes from the RIB into BGP. A route policy attached at this point is able to set any of the valid BGP attributes on the routes that are being injected.

The following example shows a route policy attached at the network attach point that sets the well-known community no-export for any routes more specific than /24:

```

route-policy NetworkControl
  if destination in (0.0.0.0/0 ge 25) then
    set community (no-export) additive
  endif
end-policy

router bgp 2
  address-family ipv4 unicast
    network 172.16.0.5/27 route-policy NetworkControl

```

Redistribute

The redistribute attach point within OSPF injects routes from other routing protocol sources into the OSPF link-state database, which is done by selecting the routes it wants to import from each protocol. It then sets the OSPF parameters of cost and metric type. The policy can control how the routes are injected into OSPF by using the **set metric-type** or **set ospf-metric** command.

The following example shows how to redistribute routes from IS-IS instance instance_10 into OSPF instance 1 using the policy OSPF-redist. The policy sets the metric type to type-2 for all redistributed routes. IS-IS routes with a tag of 10 have their cost set to 100, and IS-IS routes with a tag of 20 have their OSPF cost set to 200. Any IS-IS routes not carrying a tag of either 10 or 20 are not be redistributed into the OSPF link-state database.

```

route-policy OSPF-redist
  set metric-type type-2
  if tag eq 10 then
    set ospf cost 100
  elseif tag eq 20 then
    set ospf cost 200
  else

```

```

        drop
    endif
end-policy
router ospf 1
    redistribute isis instance_10 policy OSPF-redist
    .
    .
    .

```

Show BGP

The `show bgp attach point` allows the user to display selected BGP routes that pass the given policy. Any routes that are not dropped by the attached policy are displayed in a manner similar to the output of the `show bgp` command.

In the following example, the `show bgp route-policy` command is used to display any BGP routes carrying a MED of 5:

```

route-policy sample-display
    if med eq 5 then
        pass
    endif
end-policy
!
show bgp route-policy sample-display

```

A `show bgp policy route-policy` command also exists, which runs all routes in the RIB past the named policy as if the RIB were an outbound BGP policy. This command then displays what each route looked like before it was modified and after it was modified, as shown in the following example:

show rpl route-policy test2

```

route-policy test2
    if (destination in (10.0.0.0/8 ge 8 le 32)) then
        set med 333
    endif
end-policy
!

```

show bgp

```

BGP router identifier 10.0.0.1, local AS number 2
BGP main routing table version 11
BGP scan interval 60 secs
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
*> 10.0.0.0	10.0.1.2	10			0 3 ?
*> 10.0.0.0/9	10.0.1.2	10			0 3 ?
*> 10.0.0.0/10	10.0.1.2	10			0 3 ?
*> 10.0.0.0/11	10.0.1.2	10			0 3 ?
*> 10.1.0.0/16	10.0.1.2	10			0 3 ?
*> 10.3.30.0/24	10.0.1.2	10			0 3 ?
*> 10.3.30.128/25	10.0.1.2	10			0 3 ?
*> 10.128.0.0/9	10.0.1.2	10			0 3 ?
*> 10.255.0.0/24	10.0.101.2	1000	555		0 100 e
*> 10.255.64.0/24	10.0.101.2	1000	555		0 100 e

```
....
```

```
show bgp policy route-policy test2
```

```
10.0.0.0/8 is advertised to 10.0.101.2

Path info:
  neighbor:10.0.1.2      neighbor router id:10.0.1.2
  valid external best
Attributes after inbound policy was applied:
  next hop:10.0.1.2
  MET ORG AS
  origin:incomplete neighbor as:3 metric:10
  aspath:3
Attributes after outbound policy was applied:
  next hop:10.0.1.2
  MET ORG AS
  origin:incomplete neighbor as:3 metric:333
  aspath:2 3
...
```

Table Policy

The table policy attach point allows the user to configure traffic-index values on routes as they are installed into the global routing table. This attach point supports the BGP policy accounting feature. BGP policy accounting uses the traffic indexes that are set on the BGP routes to track various counters. This way, router operators can select different sets of BGP route attributes using the matching operations and then set different traffic indexes for each different class of route they are interested in tracking.

The following example shows how to set the traffic index to 10 in IPv4 unicast routes that originated from autonomous system 10.33. Likewise, any IPv4 unicast routes that originated from autonomous system 11.60 have their traffic index set to 11 when they are installed into the FIB. These traffic indexes are then used to count traffic being forwarded on these routes inline cards by enabling the BGP policy accounting counters on the interfaces of interest.

```
route-policy sample-table
  if as-path originates-from '10.33' then
    set traffic-index 10
  elseif as-path originates-from '11.60' then
    set traffic-index 11
  endif
end-policy

router bgp 2
  address-family ipv4 unicast
    table-policy sample-table
  .
  .
  .
```

Import

The import attach point provides control over the import of routes from the global VPN IPv4 table to a particular VPN routing and forwarding (VRF) instance.

For Layer 3 VPN networks, provider edge (PE) routers learn of VPN IPv4 routes through the Multiprotocol Internal Border Gateway Protocol (MP-iBGP) from other PE routers and automatically filters out route announcements that do not contain route targets that match any import route targets of its VRFs.

This automatic route filtering happens without RPL configuration; however, to provide more control over the import of routes in a VRF, you can configure a VRF import policy.

The following example shows how to perform matches based on a route target extended community and then sets the next hop. If the route has route target value 10:91, then the next hop is set to 172.16.0.1. If the route has route target value 11:92, then the next hop is set to 172.16.0.2. If the route has Site-of-Origin (SoO) value 10:111111 or 10:111222, then the route is dropped. All other non-matching routes are dropped.

```
route-policy bgpvrif_import
  if extcommunity rt matches-any (10:91) then
    set next-hop 172.16.0.1
  elseif extcommunity rt matches-every (11:92) then
    set next-hop 172.16.0.2
  elseif extcommunity soo matches-any (10:111111, 10:111222) then
    pass
  endif
end-policy

vrf vrf_import
  address-family ipv4 unicast
    import route-policy bgpvrif_import
  .
  .
  .
```

Export

The export attach point provides control over the export of routes from a particular VRF to a global VPN IPv4 table.

For Layer 3 VPN networks, export route targets are added to the VPN IPv4 routes when VRF IPv4 routes are converted into VPN IPv4 routes and advertised through the MP-iBGP to other PE routers (or flow from one VRF to another within a PE router).

A set of export route targets is configured with the VRF without RPL configuration; however, to set route targets conditionally, you can configure a VRF export policy.

The following example shows some match and set operations supported for the export route policy. If a route matches 172.16.1.0/24 then the route target extended community is set to 10:101, and the weight is set to 211. If the route does not match 172.16.1.0/24 but the origin of the route is egp, then the local preference is set to 212 and the route target extended community is set to 10:101. If the route does not match those specified criteria, then the route target extended community 10:111222 is added to the route. In addition, RT 10:111222 is added to the route that matches any of the previous conditions as well.

```
route-policy bgpvrif_export
  if destination in (172.16.1.0/24) then
    set extcommunity rt (10:101)
    set weight 211
  elseif origin is egp then
    set local-preference 212
    set extcommunity rt (10:101)
  endif
  set extcommunity rt (10:111222) additive
end-policy
```

```
vrf vrf-export
  address-family ipv4 unicast
    export route-policy bgpvrf-export
  .
  .
  .
```

Retain Route-Target

The retain route target attach point within BGP allows the specification of match criteria based only on route target extended community. The attach point is useful at the route reflector (RR) or at the Autonomous System Boundary Router (ASBR).

Typically, an RR has to retain all IPv4 VPN routes to peer with its PE routers. These PEs might require routers tagged with different route target IPv4 VPN routes resulting in non-scalable RRs. You can achieve scalability if you configure an RR to retain routes with a defined set of route target extended communities, and a specific set of VPNs to service.

Another reason to use this attach point is for an ASBR. ASBRs do not require that VRFs be configured, but need this configuration to retain the IPv4 VPN prefix information.

The following example shows how to configure the route policy retainer and apply it to the retain route target attach point. The route is accepted if the route contains route target extended communities 10:615, 10:6150, and 15.15.15.15:15. All other non-matching routes are dropped.

```
extcommunity-set rt rtset1
  0:615,
  10:6150,
  15.15.15.15:15
end-set

route-policy retainer
  if extcommunity rt matches-any rtset1 then
    pass
  endif
end-policy

router bgp 2
  address-family vpnv4 unicast
    retain route-target route-policy retainer
  .
  .
  .
```

Allocate-Label

The allocate-label attach point provides increased control based on various attribute match operations. This attach point is typically used in inter-AS option C to decide whether the label should be allocated or not when sending updates to the neighbor for the IPv4 labeled unicast address family. The attribute setting actions supported are for pass and drop.

Label-Mode

The label-mode attachpoint provides facility to choose label mode based on arbitrary match criteria such as prefix value, community. This attach point is typically used to set the type of label mode to per-ce or per-vrf or per-prefix based on deployment preferences. The attribute setting actions supported are for pass and drop.

Neighbor-ORF

The neighbor-orf attach point provides the filtering of incoming BGP route updates using only prefix-based matching. In addition to using this as an inbound filter, the prefixes and disposition (drop or pass) are sent to upstream neighbors as an Outbound Route Filter (ORF) to allow them to perform filtering.

The following example shows how to configure a route policy orf-preset and apply it to the neighbor ORF attach point. The prefix of the route is dropped if it matches any prefix specified in orf-preset (172.16.1.0/24, 172.16.5.0/24, 172.16.11.0/24). In addition to this inbound filtering, BGP also sends these prefix entries to the upstream neighbor with a permit or deny so that the neighbor can filter updates before sending them on to their destination.

```
prefix-set orf-preset
  172.16.1.0/24,
  172.16.5.0/24,
  172.16.11.0/24
end-set

route-policy policy-orf
  if orf prefix in orf-preset then
    drop
  endif
  if orf prefix in (172.16.3.0/24, 172.16.7.0/24, 172.16.13.0/24) then
    pass
  endif

router bgp 2
  neighbor 1.1.1.1
    remote-as 3
    address-family ipv4 unicast
      orf route-policy policy-orf
    .
    .
    .
```

Next-hop

The next-hop attach point provides increased control based on protocol and prefix-based match operations. The attach point is typically used to decide whether to act on a next-hop notification (up or down) event.

Support for next-hop tracking allows BGP to monitor reachability for routes in the Routing Information Base (RIB) that can directly affect BGP prefixes. The route policy at the BGP next-hop attach point helps limit notifications delivered to BGP for specific prefixes. The route policy is applied on RIB routes. Typically, route policies are used in conjunction with next-hop tracking to monitor non-BGP routes.

The following example shows how to configure the BGP next-hop tracking feature using a route policy to monitor static or connected routes with the prefix 10.0.0.0 and prefix length 8.

```
route-policy nxthp_policy_A
  if destination in (10.0.0.0/8) and protocol in (static, connected) then
    pass
  endif
end-policy

router bgp 2
  address-family ipv4 unicast
    nexthop route-policy nxthp_policy_A
  .
  .
```

Clear-Policy

The clear-policy attach point provides increased control based on various AS path match operations when using a **clear bgp** command. This attach point is typically used to decide whether to clear BGP flap statistics based on AS-path-based match operations.

The following example shows how to configure a route policy where the in operator evaluates to true if one or more of the regular expression matches in the set my-as-set successfully match the AS path associated with the route. If it is a match, then the **clear** command clears the associated flap statistics.

```
as-path-set my-as-set
  ios-regex '_12$',
  ios-regex '_13$'
end-set

route-policy policy_a
  if as-path in my-as-set then
    pass
  else
    drop
  endif
end-policy

clear bgp ipv4 unicast flap-statistics route-policy policy_a
```

Debug

The debug attach point provides increased control based on prefix-based match operations. This attach point is typically used to filter debug output for various BGP commands based on the prefix of the route.

The following example shows how to configure a route policy that will only pass the prefix 20.0.0.0 with prefix length 8; therefore, the debug output shows up only for that prefix.

```
route-policy policy_b
  if destination in (10.0.0.0/8) then
    pass
  else
    drop
  endif
end-policy

debug bgp update policy_b
```


BGP Attributes and Operators

This table summarizes the BGP attributes and operators per attach points.

Table 4: BGP Attributes and Operators

Attach Point	Attribute	Match	Set
aggregation	as-path	in is-local length neighbor-is originates-from passes-through unique-length	—
	as-path-length	is, ge, le, eq	—
	as-path-unique-length	is, ge, le, eq	—
	community	is-empty matches-any matches-every	set set additive delete in delete not in delete all
	destination	in	—
	extcommunity cost	—	set set additive
	local-preference	is, ge, le, eq	set
	med	is, eg, ge, le	setset +set -
	next-hop	in	set
	origin	is	set
	source	in	—
	suppress-route	—	suppress-route
weight	—	set	

Attach Point	Attribute	Match	Set
allocate-label	as-path	in is-local length neighbor-is originates-from passes-through unique-length	—
	as-path-length	is, ge, le, eq	—
	as-path-unique-length	is, ge, le, eq	—
	community	is-empty matches-any matches-every	—
	destination	in	—
	label	—	set
	local-preference	is, ge, le, eq	—
	med	is, eg, ge, le	—
	next-hop	in	—
	origin	is	—
	source	in	—
clear-policy	as-path	in is-local length neighbor-is originates-from passes-through unique-length	—
	as-path-length	is, ge, le, eq	—
	as-path-unique-length	is, ge, le, eq	—

Attach Point	Attribute	Match	Set
dampening	as-path	in is-local length neighbor-is originates-from passes-through unique-length	—
	as-path-length	is, ge, le, eq	—
	as-path-unique-length	is, ge, le, eq	—
	community	is-empty matches-any matches-every	—
	dampening	—/	set dampening
	destination	in	—
	local-preference	is, ge, le, eq	—
	med	is, eg, ge, le	—
	next-hop	in	—
	origin	is	—
source	in	—	
debug	destination	in	—
default originate	med	—	set set + set -
	rib-has-route	in	—

Attach Point	Attribute	Match	Set
neighbor-in	as-path	in is-local length NA neighbor-is originates-from passes-through unique-length	prepend prepend most-recent remove as-path private-as replace
	as-path-length	is, ge, le, eq	—
	as-path-unique-length	is, ge, le, eq	—
	communitycommunity with 'peeras'	is-empty matches-any matches-every	set set additive delete-in delete-not-in delete-all
	destination	in	—
	extcommunity cost	—	set set additive
	extcommunity rt	is-empty matches-any matches-every matches-within	set additive delete-in delete-not-in delete-all
	extcommunity soo	is-empty matches-any matches-every matches-within	—
	local-preference	is, ge, le, eq	set
	med	is, eg, ge, le	set set + set -

Attach Point	Attribute	Match	Set
	next-hop	in	set set peer address
	origin	is	set
	route-aggregated	route-aggregated	NA
	source	in	—
	weight	—	set

Attach Point	Attribute	Match	Set
neighbor-out	as-path	in is-local length — neighbor-is originates-from passes-through unique-length	prepend prepend most-recent remove as-path private-as replace
	as-path-length	is, ge, le, eq	—
	as-path-unique-length	is, ge, le, eq	—
	communitycommunity with 'peeras'	is-empty matches-any matches-every	set set additive delete-in delete-not-in delete-all
	destination	in	—
	extcommunity cost	—	set set additive
	extcommunity rt	is-empty matches-any matches-every matches-within	set additive delete-in delete-not-in delete-all
	extcommunity soo	is-empty matches-any matches-every matches-within	—
	local-preference	is, ge, le, eq	set
med	is, eg, ge, le		

Attach Point	Attribute	Match	Set
			set set + set - set max-unreachable set igp-cost
	next-hop	in	set set self
	origin	is	set
	path-type	is	—
	rd	in	—
	route-aggregated	route-aggregated	—
	source	in	—
	unsuppress-route	—	unsuppress-route
	vpn-distinguisher	—	set
neighbor-orf	orf-prefix	in	n/a

Attach Point	Attribute	Match	Set
network	as-path	—	prepend
	community	—	set set additive delete-in delete-not-in delete-all
	destination	in	—
	extcommunity cost	—	set set additive
	mpls-label	route-has-label	—
	local-preference	—	set
	med	—	set set+ set-
	next-hop	in	set
	origin	—	set
	route-type	is	—
	tag	is, ge, le, eq	—
	weight	—	set
	next-hop	destination	in
protocol		is,in	—
source		in	—

Attach Point	Attribute	Match	Set
redistribute	as-path	—	prepend
	community	—	set set additive delete in delete not in delete all
	destination	in	—
	extcommunity cost	—	setset additive
	local-preference	—	set
	med	—	set set+ set-
	next-hop	in	set
	origin	—	set
	mpls-label	route-has-label	—
	route-type	is	—
	tag	is, eq, ge, le	—
	weight	—	set
retain-rt	extcommunity rt	is-empty matches-any matches-every matches-within	—

Attach Point	Attribute	Match	Set
show	as-path	in is-local length neighbor-is originates-from passes-through unique-length	—
	as-path-length	is, ge, le, eq	—
	as-path-unique-length	is, ge, le, eq	—
	community	is-empty matches-any matches-every	—
	destination	in	—
	extcommunity rt	is-empty matches-any matches-every matches-within	—
	extcommunity soo	is-empty matches-any matches-every matches-within	—
	med	is, eg, ge, le	—
	next-hop	in	—
	origin	is	—
source	in	—	

Some BGP route attributes are inaccessible from some BGP attach points for various reasons. For example, the **set med igp-cost only** command makes sense when there is a configured igp-cost to provide a source value.

This table summarizes which operations are valid and where they are valid.

Table 5: Restricted BGP Operations by Attach Point

Command	import	export	aggregation	redistribution
prepend as-path most-recent	eBGP only	eBGP only	n/a	n/a
replace as-path	eBGP only	eBGP only	n/a	n/a
set med igp-cost	forbidden	eBGP only	forbidden	forbidden
set weight	n/a	forbidden	n/a	n/a
suppress	forbidden	forbidden	n/a	forbidden

Default-Information Originate

The default-information originate attach point allows the user to conditionally inject the default route 0.0.0.0/0 into the OSPF link-state database, which is done by evaluating the attached policy. If any routes in the local RIB pass the policy, then the default route is inserted into the link-state database.

The following example shows how to generate a default route if any of the routes that match 10.0.0.0/8 ge 8 le 25 are present in the RIB:

```

route-policy ospf-originate
  if rib-has-route in (10.0.0.0/8 ge 8 le 25) then
    pass
  endif
end-policy

router ospf 1
  default-information originate policy ospf-originate
  .
  .
  .

```

OSPF Policy Attach Points

This section describes each of the OSPF policy attach points and provides a summary of the OSPF attributes and operators.

Redistribute

The redistribute attach point within OSPF injects routes from other routing protocol sources into the OSPF link-state database, which is done by selecting the routes it wants to import from each protocol. It then sets the OSPF parameters of cost and metric type. The policy can control how the routes are injected into OSPF by using the **set metric-type** or **set ospf-metric** command.

The following example shows how to redistribute routes from IS-IS instance instance_10 into OSPF instance 1 using the policy OSPF-redist. The policy sets the metric type to type-2 for all redistributed routes. IS-IS routes with a tag of 10 have their cost set to 100, and IS-IS routes with a tag of 20 have their OSPF cost set

to 200. Any IS-IS routes not carrying a tag of either 10 or 20 are not be redistributed into the OSPF link-state database.

```
route-policy OSPF-redirect
  set metric-type type-2
  if tag eq 10 then
    set ospf cost 100
  elseif tag eq 20 then
    set ospf cost 200
  else
    drop
  endif
end-policy
router ospf 1
  redistribute isis instance_10 policy OSPF-redirect
  .
  .
  .
```

Area-in

The area-in attach point within OSPF allows you to filter inbound OSPF type-3 summary link-state advertisements (LSAs). The attach point provides prefix-based matching and hence increased control for filtering type-3 summary LSAs.

The following example shows how to configure the prefix for OSPF summary LSAs. If the prefix matches any of 10 .105.3.0/24, 10 .105.7.0/24, 10 .105.13.0/24, it is accepted. If the prefix matches any of 10 .106.3.0/24, 10 .106.7.0/24, 10 .106.13.0/24, it is dropped.

```
route-policy OSPF-area-in
  if destination in (10
.105.3.0/24, 10
.105.7.0/24, 10
.105.13.0/24) then
    drop
  endif
  if destination in (10
.106.3.0/24, 10
.106.7.0/24, 10
.106.13.0/24) then
    pass
  endif
end-policy

router ospf 1
  area 1
    route-policy OSPF-area-in in
```

Area-out

The area-out attach point within OSPF allows you to filter outbound OSPF type-3 summary LSAs. The attach point provides prefix-based matching and, hence, increased control for filtering type-3 summary LSAs.

The following example shows how to configure the prefix for OSPF summary LSAs. If the prefix matches any of 10 .105.3.0/24, 10 .105.7.0/24, 10 .105.13.0/24, it is announced. If the prefix matches any of 10 .105.3.0/24, 10 .105.7.0/24, 10 .105.13.0/24, it is dropped and not announced.

```
route-policy OSPF-area-out
```

```

    if destination in (10
.105.3.0/24, 10
.105.7.0/24, 10
.105.13.0/24) then
        drop
    endif
    if destination in (10
.105.3.0/24, 10
.105.7.0/24, 10
.105.13.0/24) then
        pass
    endif
end-policy

router ospf 1
  area 1
    route-policy OSPF-area-out out

```

OSPF Attributes and Operators

This table summarizes the OSPF attributes and operators per attach points.

Table 6: OSPF Attributes and Operators

Attach Point	Attribute	Match	Set
default-information originate	ospf-metric	—	set
	metric-type	—	set
	tag	—	set
	rib-has-route	in	—
redistribute	destination	in	—
	metric-type	—	set
	ospf-metric	—	set
	next-hop	in	—
	mpls-label	route-has-label	—
	rib-metric	is, le, ge, eq	na
	route-type	is	—
tag	is, eq, ge, le	set	
area-in	destination	in	—
area-out	destination	in	—

Attach Point	Attribute	Match	Set
spf-prefix-priority	destination	in	n/a
	spf-priority	n/a	set
	tag	is, le, ge, eq	n/a

Distribute-list in

The distribute-list in attach point within OSPF allows use of route policies to filter OSPF prefixes. The distribute-list in route-policy can be configured at OSPF instance, area, and interface levels. The route-policy used in the distribute-list in command supports match statements, "destination" and "rib-metric". The "set" commands are not supported in the route-policy.

These are examples of valid route-policies for "distribute-list in":

```
route-policy DEST
  if destination in (10.10.10.10/32) then
    drop
  else
    pass
  endif
end-policy
```

```
route-policy METRIC
  if rib-metric ge 10 and rib-metric le 19 then
    drop
  else
    pass
  endif
end-policy
```

```
prefix-set R-PFX
  10.10.10.30
end-set
```

```
route-policy R-SET
  if destination in R-PFX and rib-metric le 20 then
    pass
  else
    drop
  endif
end-policy
```

OSPFv3 Policy Attach Points

This section describes each of the OSPFv3 policy attach points and provides a summary of the OSPFv3 attributes and operators.

Redistribute

The redistribute attach point within OSPFv3 injects routes from other routing protocol sources into the OSPFv3 link-state database, which is done by selecting the route types it wants to import from each protocol. It then

sets the OSPFv3 parameters of cost and metric type. The policy can control how the routes are injected into OSPFv3 by using the **metric type** command.

The following example shows how to redistribute routes from BGP instance 15 into OSPF instance 1 using the policy OSPFv3-redirect. The policy sets the metric type to type-2 for all redistributed routes. BGP routes with a tag of 10 have their cost set to 100, and BGP routes with a tag of 20 have their OSPFv3 cost set to 200. Any BGP routes not carrying a tag of either 10 or 20 are not be redistributed into the OSPFv3 link-state database.

```

route-policy OSPFv3-redirect
  set metric-type type-2
  if tag eq 10 then
    set extcommunity cost 100
  elseif tag eq 20 then
    set extcommunity cost 200
  else
    drop
  endif
end-policy

router ospfv3 1
  redistribute bgp 15 policy OSPFv3-redirect
  .
  .
  .

```

OSPFv3 Attributes and Operators

This table summarizes the OSPFv3 attributes and operators per attach points.

Table 7: OSPFv3 Attributes and Operators

Attach Point	Attribute	Match	Set
default-information originate	ospf-metric	—	set
	metric-type	—	set
	tag	—	set
	rib-has-route	in	—
redistribute	destination	in	—
	ospf-metric	—	set
	metric-type	—	set
	route-type	is	—
	tag	is, eq, ge, le	—

IS-IS Policy Attach Points

This section describes each of the IS-IS policy attach points and provides a summary of the IS-IS attributes and operators.

Default-Information Originate

The default-information originate attach point within IS-IS allows the default route 0.0.0.0/0 to be conditionally injected into the IS-IS route database.

The following example shows how to generate an IPv4 unicast default route if any of the routes that match 10.0.0.0/8 ge 8 le 25 is present in the RIB. The cost of the IS-IS route is set to 100 and the level is set to level-1-2 on the default route that is injected into the IS-IS database.

```
route-policy isis-originate
  if rib-has-route in (10.0.0.0/8 ge 8 le 25) then
    set metric 100
    set level level-1-2
  endif
end-policy

router isis instance_10
  address-family ipv4 unicast
    default-information originate policy isis_originate
  .
```

Inter-area-propagate

The inter-area-propagate attach point within IS-IS allows the prefixes to be conditionally propagated from one level to another level within the same IS-IS instance.

The following example shows how to allow prefixes to be leaked from the level 1 LSP into the level 2 LSP if any of the prefixes match 10.0.0.0/8 ge 8 le 25.

```
route-policy isis-propagate
  if destination in (10.0.0.0/8 ge 8 le 25) then
    pass
  endif
end-policy

router isis instance_10
  address-family ipv4 unicast
    propagate level 1 into level 2 policy isis-propagate
  .
```

Inter-area-propagate

The inter-area-propagate attach point within IS-IS allows the prefixes to be conditionally propagated from one level to another level within the same IS-IS instance.

The following example shows how to allow prefixes to be leaked from the level 1 LSP into the level 2 LSP if any of the prefixes match 10.0.0.0/8 ge 8 le 25.

```
route-policy isis-propagate
  if destination in (10.0.0.0/8 ge 8 le 25) then
    pass
```



```
endif
end-policy

router isis instance_10
  address-family ipv4 unicast
    propagate level 1 into level 2 policy isis-propagate
  .
```

Nondestructive Editing of Routing Policy

The Nondestructive Editing of Routing Policy changes the default exit behavior under routing policy configuration mode to abort the configuration.

The default **exit** command acts as end-policy, end-set, or end-if. If the **exit** command is executed under route policy configuration mode, the changes are applied and configuration is updated. This destructs the existing policy. The **rpl set-exit-as-abort** command allows to overwrite the default behavior of the **exit** command under the route policy configuration mode.

Attached Policy Modification

Policies that are in use do, on occasion, need to be modified. In the traditional configuration model, a policy modification would be done by completely removing the policy and reentering re-entering it. However, this model allows for a window of time in which no policy is attached and default actions to be used, which is an opportunity for inconsistencies to exist. To close this window of opportunity, you can modify a policy in use at an attach point by respecifying it, which allows for policies that are in use to be changed, without having a window of time in which no policy is applied at the given attach point.



Note A route policy or set that is in use at an attach point cannot be removed because this removal would result in an undefined reference. An attempt to remove a route policy or set that is in use at an attach point results in an error message to the user.

Nonattached Policy Modification

As long as a given policy is not attached at an attach point, the policy is allowed to refer to nonexistent sets and policies. Configurations can be built that reference sets or policy blocks that are not yet defined, and then later those undefined policies and sets can be filled in. This method of building configurations gives much greater flexibility in policy definition. Every piece of policy you want to reference while defining a policy need not exist in the configuration. Thus, you can define a policy sample1 that references a policy sample2 using an apply statement even if the policy sample2 does not exist. Similarly, you can enter a policy statement that refers to a nonexistent set.

However, the existence of all referenced policies and sets is enforced when a policy is attached. Thus, if a user attempts to attach the policy sample1 with the reference to an undefined policy sample2 at an inbound BGP policy using the statement **neighbor 1.2.3.4 address-family ipv4 unicast policy sample1 in**, the configuration attempt is rejected because the policy sample2 does not exist.

Editing Routing Policy Configuration Elements

RPL is based on statements rather than on lines. That is, within the begin-end pair that brackets policy statements from the CLI, a new line is merely a separator, the same as a space character.

The CLI provides the means to enter and delete route policy statements. RPL provides a means to edit the contents of the policy between the begin-end brackets, using a text editor. The following text editors are available on the software for editing RPL policies:

- Nano (default)
- Emacs
- Vim

Editing Routing Policy Configuration Elements Using Emacs Editor

To edit the contents of a routing policy using the Emacs editor, use the following CLI command in `mode`:

```

edit
    route-policy
        name
    emacs

```

A copy of the route policy is copied to a temporary file and the editor is launched. After editing, save the editor buffer by using the Ctrl-X and Ctrl-S keystrokes. To save and exit the editor, use the Ctrl-X and Ctrl-C keystrokes. When you quit the editor, the buffer is committed. If there are no parse errors, the configuration is committed:

```

RP/0/# edit route-policy policy_A
-----
== MicroEMACS 3.8b () == rpl_edit.139281 ==
    if destination in (2001::/8) then
        drop
    endif
end-policy
!

== MicroEMACS 3.8b () == rpl_edit.139281 ==
Parsing.
83 bytes parsed in 1 sec (82)bytes/sec
Committing.
1 items committed in 1 sec (0)items/sec
Updating.
Updated Commit database in 1 sec

```

If there are parse errors, you are asked whether editing should continue:

```

RP/0/#edit route-policy policy_B
== MicroEMACS 3.8b () == rpl_edit.141738
route-policy policy_B
  set metric-type type_1
  if destination in (2001::/8) then
    drop
  endif
end-policy
!
== MicroEMACS 3.8b () == rpl_edit.141738 ==
Parsing.
105 bytes parsed in 1 sec (103)bytes/sec

% Syntax/Authorization errors in one or more commands.!! CONFIGURATION
FAILED DUE TO SYNTAX/AUTHORIZATION ERRORS
  set metric-type type_1
  if destination in (2001::/8) then
    drop
  endif
end-policy
!

Continue editing? [no]:

```

If you answer **yes**, the editor continues on the text buffer from where you left off. If you answer **no**, the running configuration is not changed and the editing session is ended.

Editing Routing Policy Configuration Elements Using Vim Editor

Editing elements of a routing policy with Vim (Vi IMproved) is similar to editing them with Emacs except for some feature differences such as the keystrokes to save and quit. To write to a current file and exit, use the **:wq** or **:x** or **ZZ** keystrokes. To quit and confirm, use the **:q** keystrokes. To quit and discard changes, use the **:q!** keystrokes.

You can reference detailed online documentation for Vim at this URL: <http://www.vim.org/>

Editing Routing Policy Configuration Elements Using CLI

The CLI allows you to enter and delete route policy statements. You can complete a policy configuration block by entering applicable commands such as **end-policy** or **end-set**. Alternatively, the CLI interpreter allows you to use the **exit** command to complete a policy configuration block. The **abort** command is used to discard the current policy configuration and return to mode.

Editing Routing Policy Configuration Elements Using Nano Editor

To edit the contents of a routing policy using the Nano editor, use the following CLI command in mode:

```
edit route-policy
```

```
name
```

```
nano
```

A copy of the route policy is copied to a temporary file and the editor is launched. After editing, enter Ctrl-X to save the file and exit the editor. The available editor commands are displayed on screen.

Detailed information on using the Nano editor is available at this URL: <http://www.nano-editor.org/>.

Not all Nano editor features are supported on the software.

Editing Routing Policy Language set elements Using XML

RPL supports editing set elements using XML. Entries can be appended, prepended, or deleted to an existing set without replacing it through XML.

Hierarchical Policy Conditions

The Hierarchical Policy Conditions feature enables the ability to specify a route policy within the "if" statement of another route policy. This ability enables route-policies to be applied for configurations that are based on hierarchical policies.

With the Hierarchical Policy Conditions feature, the software supports Apply Condition policies that can be used with various types of Boolean operators along with various other matching statements.

Apply Condition Policies

Apply Condition policies allow usage of a route-policy within an "if" statement of another route-policy.

Consider route-policy configurations *Parent*, *Child A*, and *Child B*:

```
route-policy Child A
  if destination in (10.10.0.0/16) then
    set local-pref 111
  endif
end-policy
!

route-policy Child B
  if as-path originates-from '222' then
    set community (333:222) additive
  endif
end-policy
!

route-policy Parent
  if apply Child A and apply Child B then
    set community (333:333) additive
  else
    set community (333:444) additive
  endif
end-policy
!
```

In the above scenarios, whenever the policy *Parent* is executed, the decision of the "if" condition in that is selected based on the result of policies *Child A* and *Child B*. The policy *Parent* is equivalent to policy *merged* as given below:

```
route-policy merged
  if destination in (10.10.0.0/16) and as-path originates-from '222' then
    set local-pref 111
    set community (333:222, 333:333) additive
  elseif destination in (10.10.0.0/16) then /*Only Policy Child A is pass */
```

```

set local-pref 111
set community (333:444) additive /*From else block */
elseif as-path originates-from '222' then /*Only Policy Child B is pass */
  set community (333:222, 333:444) additive /*From else block */
else
  set community (333:444) additive /*From else block */
endif
end-policy

```

Apply Conditions can be used with parameters and are supported on all attach points and on all clients. Hierarchical Apply Conditions can be used without any constraints on a cascaded level.

Existing route policy semantics can be expanded to include this Apply Condition:

```

Route-policy policy_name
  If apply policyA and apply policyB then
    Set med 100
  Else if not apply policyD then
    Set med 200
  Else
    Set med 300
  Endif
End-policy

```

Behavior of pass/drop/done RPL Statements for Simple Hierarchical Policies

This table describes the behavior of **pass/drop/done** RPL statements, with a possible sequence for executing the **done** statement for Simple Hierarchical Policies.

Route-policies with simple hierarchical policies	Possible done statement execution sequence	Behavior
pass	pass Continue_list	Marks the prefix as "acceptable" and continues with execution of continue_list statements.
drop	Stmts_list drop	Rejects the route immediately on hitting the drop statement and stops policy execution.
done	Stmts_list done	Accepts the route immediately on hitting the done statement and stops policy execution.
pass followed by done	pass Statement_list done	Exits immediately at the done statement with "accept route".
drop followed by done	drop Statement list done	This is an invalid scenario at execution point of time. Policy terminates execution at the drop statement itself, without going through the statement list or the done statement; the prefix will be rejected or dropped.

Behavior of pass/drop/done RPL Statements for Hierarchical Policy Conditions

This section describes the behavior of **pass/drop/done** RPL statements, with a possible sequence for executing the **done** statement for Hierarchical Policy Conditions.

Terminology for policy execution: "true-path", "false-path", and "continue-path".

```
Route-policy parent
  If apply hierarchical_policy_condition then
    TRUE-PATH      : if hierarchical_policy_condition returns TRUE then this path will
  be executed.
  Else
    FALSE-PATH     : if hierarchical_policy_condition returns FALSE then this path will
  be executed.
  End-if
  CONTINUE-PATH   : Irrespective of the TRUE/FALSE this path will be executed.
End-policy
```

Hierarchical policy conditions	Possible done statement execution sequence	Behavior
pass	pass Continue_list	Marks the return value as "true" and continues execution within the same policy condition. If there is no statement after " pass ", returns "true".
pass followed by done	pass or set action statement Stmt_list done	Marks the return value as "true" and continues execution till the done statement. Returns "true" to the apply policy condition to take "true-path".
done	Stmt_list without pass or set operation DONE	Returns " false". Condition takes "false-path".
drop	Stmt_list drop Stmt_list	The prefix is dropped or rejected.

Nested Wildcard Apply Policy

The hierarchical constructs of Routing Policy Language (RPL) allows one policy to refer to another policy. The referred or called policy is known as a child policy. The policy from which another policy is referred is called calling or parent policy. A calling or parent policy can nest multiple child policies for attachment to a common set of BGP neighbors. The nested wildcard apply policy allows wildcard (*) based apply nesting. The wildcard operation permits declaration of a generic apply statement that calls all policies that contain a specific defined set of alphanumeric characters, defined on the router.

A wildcard is specified by placing an asterisk (*) at the end of the policy name in an apply statement. Passing parameters to wildcard policy is not supported. The wildcard indicates that any value for that portion of the apply policy matches.

To illustrate nested wildcard apply policy, consider this policy hierarchy:

```
route-policy Nested_Wilcard
apply service_policy_customer*
end-policy

route-policy service_policy_customer_a
if destination in prfx_set_customer_a then
set extcommunity rt (1:1) additive
endif
end-policy

route-policy service_policy_customer_b
if destination in prfx_set_customer_b then
set extcommunity rt (1:1) additive
endif
end-policy

route-policy service_policy_customer_c
if destination in prfx_set_customer_c then
set extcommunity rt (1:1) additive
endif
end-policy
```

Here, a single parent apply statement (apply service_policy_customer*) calls (inherits) all child policies that contain the identified character string "service_policy_customer". As each child policy is defined globally, the parent dynamically nests the child policies based on the policy name. The parent is configured once and inherits each child policy on demand. There is no direct association between the parent and the child policies beyond the wildcard match statement.

VRF Import Policy Enhancement

The VRF RPL based import policy feature provides the ability to perform import operation based solely on import route-policy, by matching on route-targets (RTs) and other criteria specified within the policy. No need to explicitly configure import RTs under global VRF-address family configuration mode. If the import RTs and import route-policy is already defined, then the routes will be imported from RTs configured under import RT and then follows the route-policy attached at import route-policy.

Use the **source rt import-policy** command under VRF sub-mode of VPN address-family configuration mode to enable this feature.

Match Aggregated Route

The Match Aggregated Route feature helps to match BGP aggregated route from the non-aggregated route. BGP can aggregate a group of routes into a single prefix before sending updates to a neighbor. With Match Aggregated Route feature, route policy separates this aggregated route from other routes.

Remove Private AS in Inbound Policy

BGP appends its own as-path before sending out packets to neighbors. When a packet traverses multiple iBGP neighbors, the as-path structure will have many private autonomous systems (AS) in them. The Remove Private AS in Inbound Policy will give the capability to delete those private autonomous systems using RPL

route-policy. The **remove as-path private-as** command removes autonomous systems (AS) with AS number 64512 through 65535.



CHAPTER 5

Implementing Static Routes

Static routes are user-defined routes that cause packets moving between a source and a destination to take a specified path. Static routes can be important if the software cannot build a route to a particular destination. They are useful for specifying a gateway of last resort to which all unroutable packets are sent.

This module describes how to implement static routes.

- [Prerequisites for Implementing Static Routes, on page 273](#)
- [Restrictions for Implementing Static Routes, on page 273](#)
- [Information About Implementing Static Routes, on page 274](#)
- [How to Implement Static Routes, on page 277](#)
- [Configuration Examples, on page 284](#)
- [Configure Native UCMP for Static Routing, on page 284](#)
- [IPv4 Multicast Static Routes, on page 286](#)
- [References for Static Routes, on page 287](#)

Prerequisites for Implementing Static Routes

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Restrictions for Implementing Static Routes

These restrictions apply while implementing Static Routes:

- Static routing to an indirect next hop, (any prefix learnt through the RIB and may be more specific over the AIB), that is part of a local subnet requires configuring static routes in the global table indicating the egress interfaces as next hop. To avoid forward drop, configure static routes in the global table indicating the next-hop IP address to be the next hop.
- Generally, a route is learnt from the AIB in the global table and is installed in the FIB. However, this behavior will not be replicated to leaked prefixes. This could lead to inconsistencies in forwarding behavior.

Information About Implementing Static Routes

To implement static routes you need to understand the following concepts:

Static Route Functional Overview

Table 8: Feature History Table

Feature Name	Release Information	Feature Description
Conditional Static Anycast Route Advertisement	Release 7.3.15	<p>This feature enables you to detect core isolation and improve network convergence. The conditional installation of static route in routing table (RIB) based on object state functionality has been enhanced to use event driven notification, instead of the polling mechanism used earlier. This event driven approach allows faster detection and notification of object state changes. This feature along with BGP Neighbor address-family tracking, allows users to detect network faults quicker and take corrective action faster.</p> <p>This feature improves network convergence and reduces, and in some cases, eliminates traffic outages in the event of certain network faults, such as either link or node failures or both. Without this new feature, it will take longer to detect network faults and increase the duration of network outage.</p>

Networking devices forward packets using route information that is either manually configured or dynamically learned using a routing protocol. Static routes are manually configured and define an explicit path between two networking devices. Unlike a dynamic routing protocol, static routes are not automatically updated and must be manually reconfigured if the network topology changes. The benefits of using static routes include security and resource efficiency. Static routes use less bandwidth than dynamic routing protocols, and no CPU cycles are used to calculate and communicate routes. The main disadvantage to using static routes is the lack of automatic reconfiguration if the network topology changes.

Static routes can be redistributed into dynamic routing protocols, but routes generated by dynamic routing protocols cannot be redistributed into the static routing table. No algorithm exists to prevent the configuration of routing loops that use static routes.

Static routes are useful for smaller networks with only one path to an outside network and to provide security for a larger network for certain types of traffic or links to other networks that need more control. In general, most networks use dynamic routing protocols to communicate between networking devices but may have one or two static routes configured for special cases.

Default Administrative Distance

Static routes have a default administrative distance of 1. A low number indicates a preferred route. By default, static routes are preferred to routes learned by routing protocols. Therefore, you can configure an administrative

distance with a static route if you want the static route to be overridden by dynamic routes. For example, you could have routes installed by the Open Shortest Path First (OSPF) protocol with an administrative distance of 120. To have a static route that would be overridden by an OSPF dynamic route, specify an administrative distance greater than 120.

Directly Connected Routes

The routing table considers the static routes that point to an interface as “directly connected.” Directly connected networks are advertised by IGP routing protocols if a corresponding **interface** command is contained under the router configuration stanza of that protocol.

In directly attached static routes, only the output interface is specified. The destination is assumed to be directly attached to this interface, so the packet destination is used as the next hop address. The following example shows how to specify that all destinations with address prefix 2001:0DB8::/32 are directly reachable through interface FourHundredGigE 0/0/0/0:

```
RP/0/RP0/CPU0:router(config)# router static
RP/0/RP0/CPU0:router(config-static)# address-family ipv6 unicast
RP/0/RP0/CPU0:router(config-static-afi)# 2001:0DB8::/32 FourHundredGigE 0/0/0/0
```

Directly attached static routes are candidates for insertion in the routing table only if they refer to a valid interface; that is, an interface that is both up and has IPv4 or IPv6 enabled on it.

Recursive Static Routes

In a recursive static route, only the next hop is specified. The output interface is derived from the next hop. The following example shows how to specify that all destinations with address prefix 2001:0DB8::/32 are reachable through the host with address 2001:0DB8:3000::1:

```
RP/0/RP0/CPU0:router(config)# router static
RP/0/RP0/CPU0:router(config-static)# address-family ipv6 unicast
RP/0/RP0/CPU0:router(config-static-afi)# 2001:0DB8::/32 2001:0DB8:3000::1
```

A recursive static route is valid (that is, it is a candidate for insertion in the routing table) only when the specified next hop resolves, either directly or indirectly, to a valid output interface, provided the route does not self-recurse, and the recursion depth does not exceed the maximum IPv6 forwarding recursion depth.

A route self-recurses if it is itself used to resolve its own next hop. If a static route becomes self-recursive, RIB sends a notification to static routes to withdraw the recursive route.

Assuming a BGP route 2001:0DB8:3000::0/16 with next hop of 2001:0DB8::0104, the following static route would not be inserted into the IPv6 RIB because the BGP route next hop resolves through the static route and the static route resolves through the BGP route making it self-recursive:

```
RP/0/RP0/CPU0:router(config)# router static
RP/0/RP0/CPU0:router(config-static)# address-family ipv6 unicast
RP/0/RP0/CPU0:router(config-static-afi)# 001:0DB8::/32 2001:0DB8:3000::1
```

This static route is not inserted into the IPv6 routing table because it is self-recursive. The next hop of the static route, 2001:0DB8:3000:1, resolves through the BGP route 2001:0DB8:3000:0/16, which is itself a recursive route (that is, it only specifies a next hop). The next hop of the BGP route, 2001:0DB8::0104, resolves through the static route. Therefore, the static route would be used to resolve its own next hop.

It is not normally useful to manually configure a self-recursive static route, although it is not prohibited. However, a recursive static route that has been inserted in the routing table may become self-recursive as a result of some transient change in the network learned through a dynamic routing protocol. If this occurs, the fact that the static route has become self-recursive will be detected and it will be removed from the routing table, although not from the configuration. A subsequent network change may cause the static route to no longer be self-recursive, in which case it is re-inserted in the routing table.

Fully Specified Static Routes

In a fully specified static route, both the output interface and next hop are specified. This form of static route is used when the output interface is multiaccess and it is necessary to explicitly identify the next hop. The next hop must be directly attached to the specified output interface. The following example shows a definition of a fully specified static route:

```
RP/0/RP0/CPU0:router(config)# router static
RP/0/RP0/CPU0:router(config-static)# address-family ipv6 unicast
RP/0/RP0/CPU0:router(config-static-afi)# 2001:0DB8::/32 FourHundredGigE 0/0/0/0
2001:0DB8:3000::1
```

A fully specified route is valid (that is, a candidate for insertion into the routing table) when the specified interface, IPv4 or IPv6, is enabled and up.

Floating Static Routes

Floating static routes are static routes that are used to back up dynamic routes learned through configured routing protocols. A floating static route is configured with a higher administrative distance than the dynamic routing protocol it is backing up. As a result, the dynamic route learned through the routing protocol is always preferred to the floating static route. If the dynamic route learned through the routing protocol is lost, the floating static route is used in its place.



Note By default, static routes have smaller administrative distances than dynamic routes, so static routes are preferred to dynamic routes.

Default VRF

A static route is always associated with a VPN routing and forwarding (VRF) instance. The VRF can be the default VRF or a specified VRF. Specifying a VRF, using the `vrf vrf-name` command, allows you to enter VRF configuration mode for a specific VRF where you can configure a static route. If a VRF is not specified, a default VRF static route is configured.



Note An IPv4 or IPv6 static VRF route is the same as a static route configured for the default VRF. The IPv4 and IPV6 address families are supported in each VRF.

IPv4 and IPv6 Static VRF Routes

An IPv4 or IPv6 static VRF route is the same as a static route configured for the default VRF. The IPv4 and IPv6 address families are supported in each VRF.

Dynamic ECMP

The dynamic ECMP (equal-cost multi-path) for IGP (Interior Gateway Protocol) prefixes feature supports dynamic selection of ECMP paths ranging from 1 to 64 IGP paths. ECMP for non-recursive prefixes is dynamic. This feature enables loadbalancing support in hardware among egress links.

The dynamic ECMP (equal-cost multi-path) for IGP (Interior Gateway Protocol) prefixes feature supports dynamic selection of ECMP paths ranging from 1 to 64 IGP paths. ECMP for non-recursive prefixes is dynamic.

This feature enables loadbalancing support in hardware among egress links.

How to Implement Static Routes

This section contains the following procedures:

Configure Static Route

Static routes are entirely user configurable and can point to a next-hop interface, next-hop IP address, or both. In the software, if an interface was specified, then the static route is installed in the Routing Information Base (RIB) if the interface is reachable. If an interface was not specified, the route is installed if the next-hop address is reachable. The only exception to this configuration is when a static route is configured with the permanent attribute, in which case it is installed in RIB regardless of reachability.



Tip You can programmatically configure and view the operational state of the static route using `openconfig-local-routing.yang` or `openconfig-network-instance.yang` OpenConfig data models. To get started with using data models, see the .

This task explains how to configure a static route.

Step 1 **configure**

Example:

```
RP/0/# configure
Enters mode.
```

Step 2 **router static**

Example:

```
RP/0/RP0/CPU0:router(config)# router static
```

Enters static route configuration mode.

Step 3 `vrf vrf-name`

Example:

```
RP/0/RP0/CPU0:router(config-static)# vrf vrf_A
```

(Optional) Enters VRF configuration mode.

If a VRF is not specified, the static route is configured under the default VRF.

Step 4 `address-family { ipv4 | ipv6 } { unicast | multicast }`

Example:

```
RP/0/RP0/CPU0:router(config-static-vrf)# address family ipv4 unicast
```

Enters address family mode.

Step 5 `prefix mask [vrf vrf-name] { ip-address | interface-type interface-instance } [distance] [description text] [tag tag] [permanent]`

Example:

```
RP/0/RP0/CPU0:router(config-static-vrf-afi)# 10.0.0.0/8 172.20.16.6 110
```

Configures an administrative distance of 110.

- This example shows how to route packets for network 10.0.0.0 through to a next hop at 172.20.16.6 if dynamic information with administrative distance less than 110 is not available.

Step 6 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

A default static route is often used in simple router topologies. In the following example, a route is configured with an administrative distance of 110.

```
configure
router static
address-family ipv4 unicast
0.0.0.0/0 2.6.0.1 110
end
```

Configure Floating Static Route

This task explains how to configure a floating static route.

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 **router static**

Example:

```
RP/0/RP0/CPU0:router(config)# router static
```

Enters static route configuration mode.

Step 3 **vrf vrf-name**

Example:

```
RP/0/RP0/CPU0:router(config-static)# vrf vrf_A
```

(Optional) Enters VRF configuration mode.

If a VRF is not specified, the static route is configured under the default VRF.

Step 4 **address-family { ipv4 | ipv6 } { unicast | multicast }**

Example:

```
RP/0/RP0/CPU0:router(config-static-vrf)# address family ipv6 unicast
```

Enters address family mode.

Step 5 **prefix mask [vrf vrf-name] { ip-address | interface-type interface-instance } [distance] [description text] [tag tag] [permanent]**

Example:

```
RP/0/RP0/CPU0:router(config-static-vrf-afi)# 2001:0DB8::/32 2001:0DB8:3000::1 201
```

Configures an administrative distance of 201.

Step 6 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.

- **Cancel** —Remains in the configuration session, without committing the configuration changes.

A floating static route is often used to provide a backup path if connectivity fails. In the following example, a route is configured with an administrative distance of 201.

```
configure
router static
address-family ipv6 unicast
2001:0DB8::/32 2001:0DB8:3000::1 201
end
```

Configure Static Routes Between PE-CE Routers

This task explains how to configure static routing between PE-CE routers.



Note VRF fallback is not supported with IPv6 VPN Provider Edge (6VPE).

Step 1 **configure**

Example:

```
RP/0/# configure
Enters mode.
```

Step 2 **vrf vrf-name**

Example:

```
RP/0/RP0/CPU0:router(config)# router vrf_A
Enters VRF configuration mode.
```

Step 3 **address-family { ipv4 | ipv6 } { unicast | multicast }**

Example:

```
RP/0/RP0/CPU0:router(config-vrf)# address family ipv6 unicast
Enters address family mode.
```

Step 4 **exit**

Example:

```
RP/0/RP0/CPU0:router(config-vrf)# exit
Exits VRF configuration mode.
```


Step 5 `router static`**Example:**

```
RP/0/RP0/CPU0:router(config)# router static
```

Enters static route configuration mode.

Step 6 `vrf vrf-name`**Example:**

```
RP/0/RP0/CPU0:router(config-static)# vrf vrf_A
```

(Optional) Enters VRF configuration mode.

If a VRF is not specified, the static route is configured under the default VRF.

Step 7 `address-family { ipv4 | ipv6 } { unicast | multicast }`**Example:**

```
RP/0/RP0/CPU0:router(config-static-vrf)# address family ipv6 unicast
```

Enters address family mode.

Step 8 `prefix mask [vrf vrf-name] { ip-address | interface-type interface- path-id } [distance] [description text] [tag tag] [permanent]`**Example:**

```
RP/0/RP0/CPU0:router(config-static-vrf-afi)# 2001:0DB8::/32 2001:0DB8:3000::1 201
```

Configures an administrative distance of 201.

Step 9 Use the `commit` or `end` command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

In the following example, a static route between PE and CE routers is configured, and a VRF is associated with the static route:

```
configure
router static
vrf vrf_A
address-family ipv4 unicast
0.0.0.0/0 2.6.0.2 120
end
```

Change Maximum Number of Allowable Static Routes

This task explains how to change the maximum number of allowable static routes.

Before you begin



Note The number of static routes that can be configured on a router for a given address family is limited by default to 4000. The limit can be raised or lowered using the **maximum path** command. Note that if you use the **maximum path** command to reduce the configured maximum allowed number of static routes for a given address family below the number of static routes currently configured, the change is rejected. In addition, understand the following behavior: If you commit a batch of routes that would, when grouped, push the number of static routes configured above the maximum allowed, the first *n* routes in the batch are accepted. The number previously configured is accepted, and the remainder are rejected. The *n* argument is the difference between the maximum number allowed and number previously configured.

Step 1 configure

Example:

```
RP/0/# configure
Enters mode.
```

Step 2 router static

Example:

```
RP/0/(config)# router static
Enters static route configuration mode.
```

Step 3 maximum path { ipv4 | ipv6 } value

Example:

```
RP/0/(config-static)# maximum path ipv4 10000
Changes the maximum number of allowable static routes.
```

- Specify IPv4 or IPv6 address prefixes.
- Specify the maximum number of static routes for the given address family. The range is from 1 to 140000.
- This example sets the maximum number of static IPv4 routes to 10000.

Step 4 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.

- **Cancel**—Remains in the configuration session, without committing the configuration changes.

Configuring a static route to point at interface null 0 may be used for discarding traffic to a particular prefix. For example, if it is required to discard all traffic to prefix 2001:0DB8:42:1/64, the following static route would be defined:

```
configure
router static
address-family ipv6 unicast
2001:0DB8:42:1::/64 null 0
end
```

Associate VRF with a Static Route

This task explains how to associate a VRF with a static route.

Step 1 **configure**

Example:

```
RP/0/# configure
Enters mode.
```

Step 2 **router static**

Example:

```
RP/0/
/CPU0:router(config)# router static
Enters static route configuration mode.
```

Step 3 **vrf vrf-name**

Example:

```
RP/0/RP0/CPU0:router(config-static)# vrf vrf_A
Enters VRF configuration mode.
```

Step 4 **address-family { ipv4 | ipv6 } { unicast | multicast }**

Example:

```
RP/0/RP0/CPU0:router(config-static-vrf)# address family ipv6 unicast
Enters address family mode.
```

Step 5 **prefix mask [vrf vrf-name] {next-hop ip-address | interface-name} {path-id} [distance] [description text] [tag tag] [permanent]**

Example:

```
RP/0/RP0/CPU0:router(config-static-vrf-afi)# 2001:0DB8::/32 2001:0DB8:3000::1 201
```

Configures an administrative distance of 201.

Step 6 `interface interface-name`**Example:**

```
RP/0/  
/CPU0:router(config)# interface FourHundredGigE 0/5/0/0
```

Enters interface configuration mode.

Note Ensure that the name of the interface you configured in this step is the same as the VRF name configured in step 5.

Step 7 Use the `commit` or `end` command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuration Examples

This section provides the following configuration examples:

Configure Native UCMP for Static Routing

In a network where traffic is load balanced on two or more links, configuring equal metrics on the links would create Equal Cost Multipath (ECMP) next hops. Because the bandwidth of the links is not taken into consideration while load balancing, the higher bandwidth links are underutilized. To avoid this problem, you can configure Unequal Cost Multipath (UCMP), either locally (local UCMP), or natively (native UCMP) so that the higher bandwidth links carry traffic in proportion to the capacity of the links. UCMP supports IPv4 and IPv6 static VRF routes.

Local UCMP: All static routes are configured with the same link metrics. The static IGP calculates the load metric based on the bandwidth of the links and load balances the traffic across the links. However, local UCMP does not consider bandwidth while load balancing across links that are closer to the destination (multiple hops away).

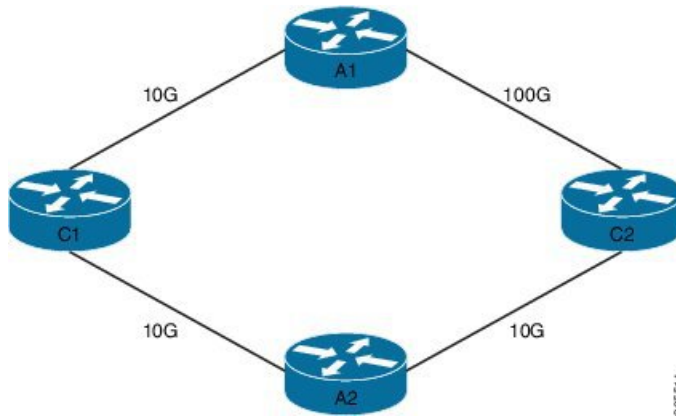
Native UCMP: Static routes over higher bandwidth links are configured with lower link metrics so that they are preferred to routes over lower bandwidth links. The static IGP calculates the load metric based on the bandwidth of the links and determines the percentage of traffic going out of the higher and lower bandwidth

links. By matching the configured link metrics with end-to-end available bandwidth, native UCMP is able to effectively load balance traffic across links that are closer to the destination (multiple hops away).

Configuration Example

Consider the topology in the following figure. For load balancing traffic out of Router A1, if local UCMP is used, then both 10G and 100G links will have equal link metrics. The static IGP decides to send more traffic out of the 100G link because of the higher load metric. However, for load balancing traffic out of Router A2, local UCMP works only on links to Routers C1 and C2. For load balancing traffic from Router C1 to Router A1 and Router C2 to Router A1, native UCMP is preferred. As a result, local UCMP is used only on single hop destinations, and native UCMP is used for multi-hop destinations.

Figure 8: Unequal Cost Multipath for Static Routing



To configure UCMP for static routing, use the following steps:

1. Enter the global configuration mode.

```
RP/0/0/CPU0:Router# configure
```

2. Enter the static routing mode.

```
RP/0/0/CPU0:Router(config)# router static
```

3. Configure UCMP with load metric for IPv4 or IPv6 address family.

```
RP/0/0/CPU0:Router(config-static)# address-family ipv4 unicast
RP/0/0/CPU0:Router(config-static-afi)# 10.10.10.1/32 FourHundredGigE 0/5/0/0 metric 10
```

In this example, we have configured UCMP for IPv4 address family. To configure UCMP for IPv6 address family, use the following sample configuration.

```
RP/0/0/CPU0:Router(config-static)# address-family ipv6 unicast
RP/0/0/CPU0:Router(config-static-afi)# 10:10::1/64 FourHundredGigE0/5/0/0 metric 10
```

4. Exit the static configuration mode and commit your configuration.

```
RP/0/0/CPU0:Router(config-static-afi)# exit
RP/0/0/CPU0:Router(config-static)# exit
RP/0/0/CPU0:Router(config)# commit
Fri Feb 19 06:16:33.164 IST
RP/0/0/CPU0:Feb 19 06:16:34.273 : ipv4_static[1044]:
%ROUTING-IP_STATIC-4-CONFIG_NEXTHOP_ETHER_INTERFACE :
Route for 10.10.10.1 is configured via ethernet interface
```

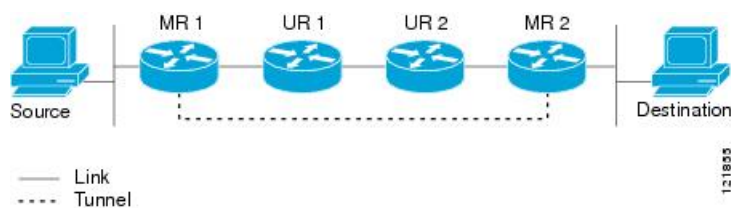
Repeat this procedure on all routers that need to be configured with UCMP.

IPv4 Multicast Static Routes

IP multicast static routes (mroutes) allow you to have multicast paths diverge from the unicast paths. When using Protocol Independent Multicast (PIM), the router expects to receive packets on the same interface where it sends unicast packets back to the source. This expectation is beneficial if your multicast and unicast topologies are congruent. However, you might want unicast packets to take one path and multicast packets to take another.

The most common reason for using separate unicast and multicast paths is tunneling. When a path between a source and a destination does not support multicast routing, configuring two routers with a GRE tunnel between them is the solution. In the figure below, each unicast router (UR) supports unicast packets only; each multicast router (MR) supports multicast packets.

Figure 9: Tunnel for Multicast Packets



In the figure, the source delivers multicast packets to destination by using MR 1 and MR 2. MR 2 accepts the multicast packet only if it predicts it can reach source over the tunnel. If this situation is true, when the destination sends unicast packets to the source, MR 2 sends them over the tunnel. The check that MR2 can reach the source over the tunnel is a Reverse Path Forwarding (RPF) check, and the static mroute allows the check to be successful when the interface, on which the multicast packet arrives, is not the unicast path back to the source. Sending the packet over the tunnel could be slower than natively sending it through UR 2, UR 1, and MR 1.

A multicast static route allows you to use the configuration in the above figure by configuring a static multicast source. The system uses the configuration information instead of the unicast routing table to route the traffic. Therefore, multicast packets can use the tunnel without having the unicast packets use the tunnel. Static mroutes are local to the router they are configured on and not advertised or redistributed in any way to any other router.

Configure Multicast Static Routes

The following example shows how to configure multiple static routes in IPv4 and IPv6 address family configuration modes:

```
/* Enables a static routing process */
Router(config)# router static

/* Configures the IPv4 address-family for the unicast topology with a destination prefix.
*/
Router(config-static)# address-family ipv4 unicast
Router(config-static-afi)# 10.1.1.0/24 198.51.100.1
Router(config-static-afi)# 223.255.254.254/32 203.0.113.1
Router(config-static-afi)# exit

/* Configures the IPv4 address-family for the multicast topology with a destination prefix.
*/
Router(config-static)# address-family ipv4 multicast
```

```

Router(config-static-afi)# 198.51.100.20/32 209.165.201.0
Router(config-static-afi)# 192.0.2.10/32 209.165.201.0
Router(config-static-afi)# exit

/* Enable the address family IPv4 and IPv6 multicast on the next hop interface. */
Router(config)# interface FourHundredGigE 0/0/0/12
Router(config-if)# address-family ipv4 multicast
Router(config-if)# address-family ipv6 multicast

```

Running Configuration

```

router static
 address-family ipv4 unicast
  10.1.1.0/24 198.51.100.1
  223.255.254.254/32 203.0.113.1
  !
 address-family ipv4 multicast
  198.51.100.20/32 209.165.201.0
  192.0.2.10/32 209.165.201.0
  !
 interface FourHundredGigE 0/0/0/12
  address-family ipv4 multicast
  address-family ipv6 multicast

```

Verification

Verify the IPv4 multicast routes.

```
show route ipv4 multicast
```

```

Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
U - per-user static route, o - ODR, L - local, G - DAGR, l - LISp
A - access/subscriber, a - Application route
M - mobile route, r - RPL, t - Traffic Engineering, (!) - FRR Backup path

Gateway of last resort is 10.1.1.20 to network 0.0.0.0

i*L1 0.0.0.0/0 [115/10] via 10.1.1.20, 00:41:12, FourHundredGigE0/0/0/6
C 10.1.1.0/24 is directly connected, 00:41:12, FourHundredGigE0/0/0/0
L 10.1.1.10/32 is directly connected, 00:41:12, FourHundredGigE0/0/0/0
S 172.16.2.10/32 [1/0] via 198.51.100.20, 00:41:12
i L1 172.16.3.1/32 [115/20] via 198.51.100.20, 00:41:12, FourHundredGigE0/0/0/12
i L1 192.0.2.1/24 [115/20] via 198.51.100.20, 00:41:12, FourHundredGigE0/0/0/1

```

References for Static Routes

The following topics provide additional conceptual information on static routes:

- [Static Route Functional Overview, on page 274](#)
- [Default Administrative Distance, on page 274](#)

- [Directly Connected Routes](#), on page 275
- [Floating Static Routes](#), on page 276
- [Fully Specified Static Routes](#), on page 276
- [Recursive Static Routes](#), on page 275

Static Route Functional Overview

Table 9: Feature History Table

Feature Name	Release Information	Feature Description
Conditional Static Anycast Route Advertisement	Release 7.3.15	<p>This feature enables you to detect core isolation and improve network convergence. The conditional installation of static route in routing table (RIB) based on object state functionality has been enhanced to use event driven notification, instead of the polling mechanism used earlier. This event driven approach allows faster detection and notification of object state changes. This feature along with BGP Neighbor address-family tracking, allows users to detect network faults quicker and take corrective action faster.</p> <p>This feature improves network convergence and reduces, and in some cases, eliminates traffic outages in the event of certain network faults, such as either link or node failures or both. Without this new feature, it will take longer to detect network faults and increase the duration of network outage.</p>

Networking devices forward packets using route information that is either manually configured or dynamically learned using a routing protocol. Static routes are manually configured and define an explicit path between two networking devices. Unlike a dynamic routing protocol, static routes are not automatically updated and must be manually reconfigured if the network topology changes. The benefits of using static routes include security and resource efficiency. Static routes use less bandwidth than dynamic routing protocols, and no CPU cycles are used to calculate and communicate routes. The main disadvantage to using static routes is the lack of automatic reconfiguration if the network topology changes.

Static routes can be redistributed into dynamic routing protocols, but routes generated by dynamic routing protocols cannot be redistributed into the static routing table. No algorithm exists to prevent the configuration of routing loops that use static routes.

Static routes are useful for smaller networks with only one path to an outside network and to provide security for a larger network for certain types of traffic or links to other networks that need more control. In general, most networks use dynamic routing protocols to communicate between networking devices but may have one or two static routes configured for special cases.

Default Administrative Distance

Static routes have a default administrative distance of 1. A low number indicates a preferred route. By default, static routes are preferred to routes learned by routing protocols. Therefore, you can configure an administrative distance with a static route if you want the static route to be overridden by dynamic routes. For example, you could have routes installed by the Open Shortest Path First (OSPF) protocol with an administrative distance of 120. To have a static route that would be overridden by an OSPF dynamic route, specify an administrative distance greater than 120.

Directly Connected Routes

The routing table considers the static routes that point to an interface as “directly connected.” Directly connected networks are advertised by IGP routing protocols if a corresponding **interface** command is contained under the router configuration stanza of that protocol.

In directly attached static routes, only the output interface is specified. The destination is assumed to be directly attached to this interface, so the packet destination is used as the next hop address. The following example shows how to specify that all destinations with address prefix 2001:0DB8::/32 are directly reachable through interface FourHundredGigE 0/0/0/0:

```
RP/0/RP0/CPU0:router(config)# router static
RP/0/RP0/CPU0:router(config-static)# address-family ipv6 unicast
RP/0/RP0/CPU0:router(config-static-afi)# 2001:0DB8::/32 FourHundredGigE 0/0/0/0
```

Directly attached static routes are candidates for insertion in the routing table only if they refer to a valid interface; that is, an interface that is both up and has IPv4 or IPv6 enabled on it.

Floating Static Routes

Floating static routes are static routes that are used to back up dynamic routes learned through configured routing protocols. A floating static route is configured with a higher administrative distance than the dynamic routing protocol it is backing up. As a result, the dynamic route learned through the routing protocol is always preferred to the floating static route. If the dynamic route learned through the routing protocol is lost, the floating static route is used in its place.



Note By default, static routes have smaller administrative distances than dynamic routes, so static routes are preferred to dynamic routes.

Fully Specified Static Routes

In a fully specified static route, both the output interface and next hop are specified. This form of static route is used when the output interface is multiaccess and it is necessary to explicitly identify the next hop. The next hop must be directly attached to the specified output interface. The following example shows a definition of a fully specified static route:

```
RP/0/RP0/CPU0:router(config)# router static
RP/0/RP0/CPU0:router(config-static)# address-family ipv6 unicast
RP/0/RP0/CPU0:router(config-static-afi)# 2001:0DB8::/32 FourHundredGigE 0/0/0/0
2001:0DB8:3000::1
```

A fully specified route is valid (that is, a candidate for insertion into the routing table) when the specified interface, IPv4 or IPv6, is enabled and up.

Recursive Static Routes

In a recursive static route, only the next hop is specified. The output interface is derived from the next hop. The following example shows how to specify that all destinations with address prefix 2001:0DB8::/32 are reachable through the host with address 2001:0DB8:3000::1:

```
RP/0/RP0/CPU0:router(config)# router static
RP/0/RP0/CPU0:router(config-static)# address-family ipv6 unicast
RP/0/RP0/CPU0:router(config-static-afi)# 2001:0DB8::/32 2001:0DB8:3000::1
```

A recursive static route is valid (that is, it is a candidate for insertion in the routing table) only when the specified next hop resolves, either directly or indirectly, to a valid output interface, provided the route does not self-recurse, and the recursion depth does not exceed the maximum IPv6 forwarding recursion depth.

A route self-recurses if it is itself used to resolve its own next hop. If a static route becomes self-recursive, RIB sends a notification to static routes to withdraw the recursive route.

Assuming a BGP route 2001:0DB8:3000::0/16 with next hop of 2001:0DB8::0104, the following static route would not be inserted into the IPv6 RIB because the BGP route next hop resolves through the static route and the static route resolves through the BGP route making it self-recursive:

```
RP/0/RP0/CPU0:router(config)# router static
RP/0/RP0/CPU0:router(config-static)# address-family ipv6 unicast
RP/0/RP0/CPU0:router(config-static-afi)# 001:0DB8::/32 2001:0DB8:3000::1
```

This static route is not inserted into the IPv6 routing table because it is self-recursive. The next hop of the static route, 2001:0DB8:3000:1, resolves through the BGP route 2001:0DB8:3000:0/16, which is itself a recursive route (that is, it only specifies a next hop). The next hop of the BGP route, 2001:0DB8::0104, resolves through the static route. Therefore, the static route would be used to resolve its own next hop.

It is not normally useful to manually configure a self-recursive static route, although it is not prohibited. However, a recursive static route that has been inserted in the routing table may become self-recursive as a result of some transient change in the network learned through a dynamic routing protocol. If this occurs, the fact that the static route has become self-recursive will be detected and it will be removed from the routing table, although not from the configuration. A subsequent network change may cause the static route to no longer be self-recursive, in which case it is re-inserted in the routing table.



CHAPTER 6

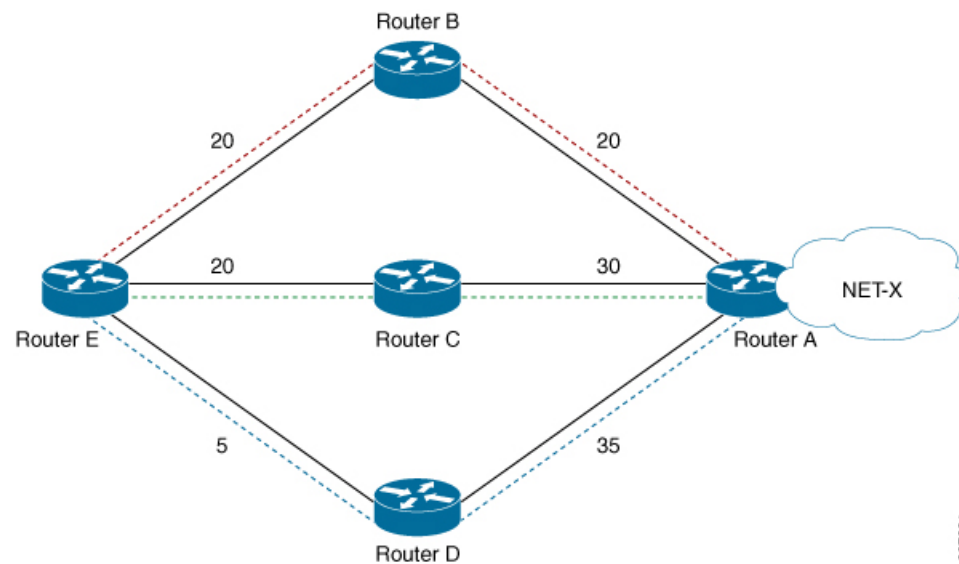
Implementing UCMP

The unequal cost multipath (UCMP) load-balancing provides the capability to load balance traffic proportionally across multiple paths, with different cost. Generally, higher bandwidth paths have lower Interior Gateway Protocol (IGP) metrics configured, so that they form the shortest IGP paths.

With the UCMP load-balancing enabled, protocols can use even lower bandwidth paths or higher cost paths for traffic, and can install these paths to the forwarding information base (FIB). These protocols still install multiple paths to the same destination in FIB, but each path will have a 'load metric/weight' associated with it. FIB uses this load metric/weight to decide the amount of traffic that needs to be sent on a higher bandwidth path and the amount of traffic that needs to be sent on a lower bandwidth path.

In the following example, there are 3 paths to get to Network X as follows:

Figure 10: Topology for UCMP



Paths	Cost from Router E to Net -X
E-B-A	40
E-C-A	50
E-D-A	40

IGP selects the lowest path links, i.e E-B-A and E-D-A. The path E-C-A is not considered for load balancing because of higher cost. The lowest path link E-D (5) is not a tie breaker, as the end to end cost to the Network X is considered.

- [ECMP vs. UCMP Load Balancing, on page 292](#)
- [UCMP Minimum Integer Ratio, on page 292](#)
- [Configuring IS-IS With Weight, on page 293](#)
- [Configuring IS-IS With Metric, on page 294](#)
- [Configuring BGP With Weights, on page 295](#)

ECMP vs. UCMP Load Balancing

Load balancing is a forwarding mechanism that distributes traffic over multiple links based on certain parameters. Equal Cost Multi Path (ECMP) is a forwarding mechanism for routing packets along multiple paths of equal cost with the goal to achieve almost equally distributed link load sharing. This significantly impacts a router's next-hop (path) decision.

In ECMP, it is assumed that all links available are of similar speed which inherently means that the hash values that are computed are equally shared over the multiple paths available.

For instance, if we have two paths available, the buckets (which in the end identify the links to be chosen) will be assigned in a 50% / 50% loadsharing. This can be problematic when one path is say a 10G link and the other link is a 1G link. In this case, you probably want to assign a (near) 90/10 type deviation, but considering that BGP is not bandwidth aware, the 10G path is still chosen 50% of the time as much as the 1G link. In this scenario, not all paths are of equal cost path.

What UCMP does in this case is apply a *weight* to a path which means that we are giving more hash buckets to one path that has a higher weight. The weight applied is *static* in the sense that it is derived by the DMZ bandwidth extended community either assigned to a peer or as configured via the Route Policy Language (RPL) route manipulation functionality.

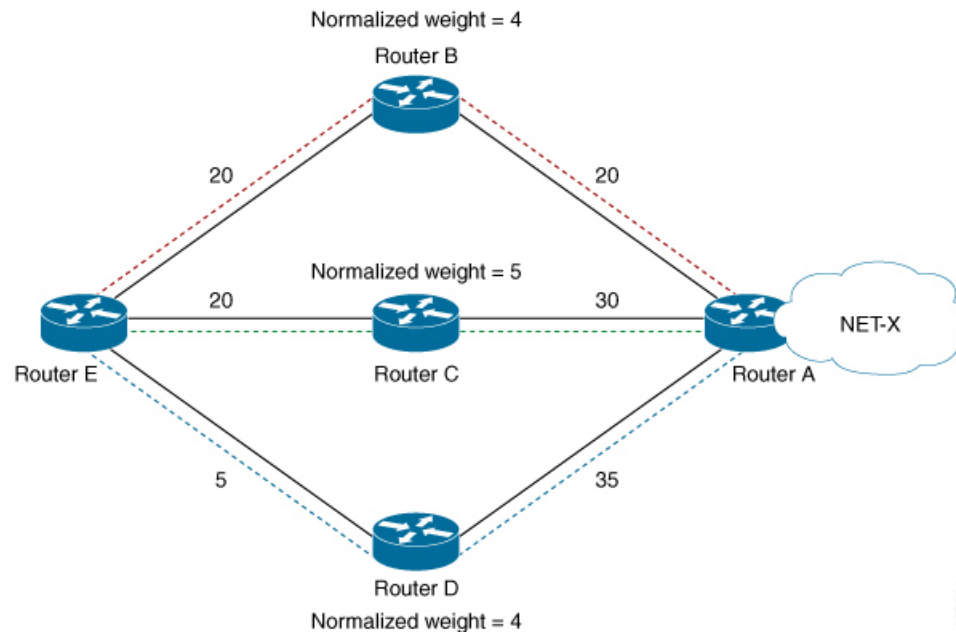
In general, a routing protocol decides a best path to a destination based on a metric. This metric is generally driven by the bandwidth of the circuit. When we have 3 paths available, say 1G/10G/100G, routing protocols generally discard the 1G/10G paths available. In defined cases, one may want to spread the load over the circuits based on the load they can carry. In this example, one may want to distribute traffic in a 1%/10%/89% fashion over the 1G/10G/100G paths available.

UCMP Minimum Integer Ratio

The UCMP Minimum Integer Ratio feature saves hardware resources when programming UCMP, by using optimized number of buckets.

To calculate the UCMP minimum integer ratio, find the greatest common divisor (GCD) and divide all the calculated normalized weights.

In the following Figure, we have three configured weights 40, 50, and 40, with GCD as 10. To calculate the normalized weight, divide the configured weight by GCD. In this example, we need to divide 40 by 10, 50 by 10, and 40 by 10, which is 4, 5, and 4 respectively. Therefore 4, 5, and 4 are the new normalized weights.



New normalized weights are: $40/10 = 4$, $50/10 = 5$, and $40/10 = 4$

If GCD is 1, then Normalized Weight = $(\text{Path weight}/\text{Total weight}) * \text{Maximum bucket size}$

Configuring IS-IS With Weight

The following example shows the IS-IS weight configuration with IPv4. The same can be done for IPv6, with or without SR.

```
CPU0:router(config)# router isis 1
RP/0/RSP0/CPU0:router(config-isis)# interface HundredGigE 0/3/0/8
RP/0/RSP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-isis-if-af)# weight 200
RP/0/RSP0/CPU0:router(config-isis)# interface HundredGigE 0/3/0/9
RP/0/RSP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-isis-if-af)# weight 300
```

Verification

The following example verifies CEF entry. Then, for two paths with weights of 200 and 300 respectively, and GCD of 100; the expected normalized weights are 2 and 3.

```
Router# show cef ipv4 97.0.0.0 detail
```

```
97.0.0.0/24, version 537, internal 0x1000001 0x0 (ptr 0x71bcaee0) [1], 0x0 (0x71b98870),
0x0 (0x0)
Updated Oct 16 06:34:46.197
remote adjacency to HundredGigE 0/3/0/8
Prefix Len 24, traffic index 0, precedence n/a, priority 2
gateway array (0x71a6de10) reference count 13, flags 0x0, source rib (7), 0 backups
[14 type 3 flags 0x8401 (0x71b02d90) ext 0x0 (0x0)]
LW-LDI[type=3, refc=1, ptr=0x71b98870, sh-ldi=0x71b02d90]
gateway array update type-time 1 Oct 16 06:34:46.196
LDI Update time Oct 16 06:34:46.197
```

```

LW-LDI-TS Oct 16 06:34:46.197
  via 1.0.0.2/32, HundredGigE0/3/0/8, 4 dependencies, weight 200, class 0 [flags 0x0]
    path-idx 0 NHID 0x0 [0x7244d2a4 0x0]
    next hop 1.0.0.2/32
    remote adjacency
  via 2.0.0.2/32, HundredGigE0/3/0/9, 4 dependencies, weight 300, class 0 [flags 0x0]
    path-idx 1 NHID 0x0 [0x7244d2f8 0x0]
    next hop 2.0.0.2/32
    remote adjacency

Weight distribution:
slot 0, weight 200, normalized_weight 2, class 0
slot 1, weight 300, normalized_weight 3, class 0

Load distribution: 0 1 0 1 1 (refcount 14)

Hash  OK  Interface                Address
0      Y   HundredGigE0/3/0/8      remote
1      Y   HundredGigE0/3/0/9      remote
2      Y   HundredGigE0/3/0/8      remote
3      Y   HundredGigE0/3/0/9      remote
4      Y   HundredGigE0/3/0/9      remote

```

Configuring IS-IS With Metric

The following example shows IS-IS metric configuration with IPv4. The same can be done with IPv6.

```

Router# enable
RP/0/RSP0/CPU0:router(config)# router isis 1
RP/0/RSP0/CPU0:router(config-isis)# interface HundredGigE0/3/0/8
RP/0/RSP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-isis-if-af)# metric 1
RP/0/RSP0/CPU0:router(config-isis)# interface HundredGigE0/3/0/9
RP/0/RSP0/CPU0:router(config-isis-if)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-isis-if-af)# metric 100

```

Verification

The following example verifies CEF entry, and checks for the two paths with metric values of 1 and 100, respectively. In this example, the best path route metric is 21 and the UCMP path route metric is 120. Therefore, the calculation is as follows:

The best path route metric, 21 = (1 configured + 20 added by IS-IS), weight 0xFFFFFFFF (4294967295)

The UCMP path route metric, 120 = (100 + 20), weight = (21/120) * 4294967295 = 751619276

GCD is one. So Normalized Weight is:

$$(4294967295 * 64) / (4294967295 + 751619276) = 54$$

$$(751619276 * 64) / (4294967295 + 751619276) = 9$$

```
Router# show cef ipv4 97.0.0.0 detail
```

```

97.0.0.0/24, version 773, internal 0x1000001 0x0 (ptr 0x71bcaee0) [1], 0x0 (0x71b98870),
0x0 (0x0)
Updated Oct 16 06:36:08.632
remote adjacency to HundredGigE0/3/0/8
Prefix Len 24, traffic index 0, precedence n/a, priority 2
gateway array (0x71a6d9d0) reference count 2, flags 0x0, source rib (7), 0 backups

```

```

[3 type 3 flags 0x8401 (0x71b02b90) ext 0x0 (0x0)]
LW-LDI [type=3, refc=1, ptr=0x71b98870, sh-ldi=0x71b02b90]
gateway array update type-time 1 Oct 16 06:36:08.632
LDI Update time Oct 16 06:36:08.632
LW-LDI-TS Oct 16 06:36:08.632
  via 1.0.0.2/32, HundredGigE0/3/0/8, 14 dependencies, weight 4294967295, class 0 [flags
0x0]
    path-idx 0 NHID 0x0 [0x7244d2a4 0x0]
    next hop 1.0.0.2/32
    remote adjacency
  via 2.0.0.2/32, HundredGigE0/3/0/9, 14 dependencies, weight 751619276, class 0 [flags
0x0]
    path-idx 1 NHID 0x0 [0x7244d2f8 0x0]
    next hop 2.0.0.2/32
    remote adjacency

Weight distribution:
slot 0, weight 4294967295, normalized_weight 54, class 0
slot 1, weight 751619276, normalized_weight 9, class 0

```

Configuring BGP With Weights

The following example shows BGP configuration with weights.

```

RP/0/RSP0/CPU0:router(config)# route-policy BW1
RP/0/RSP0/CPU0:router(config-rpl)# set extcommunity bandwidth (2906:45750000)
RP/0/RSP0/CPU0:router(config-rpl)# end-policy
RP/0/RSP0/CPU0:router(config)# !
RP/0/RSP0/CPU0:router(config)# route-policy BW2
RP/0/RSP0/CPU0:router(config-rpl)# set extcommunity bandwidth (2906:47250000)
RP/0/RSP0/CPU0:router(config-rpl)# end-policy
RP/0/RSP0/CPU0:router(config)# !
RP/0/RSP0/CPU0:router(config)# route-policy pass-all
RP/0/RSP0/CPU0:router(config-rpl)# pass
RP/0/RSP0/CPU0:router(config-rpl)# end-policy
RP/0/RSP0/CPU0:router(config)# !
RP/0/RSP0/CPU0:router(config)# router bgp 1
RP/0/RSP0/CPU0:router(config-bgp)# bgp bestpath as-path multipath-relax
RP/0/RSP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-bgp-af)# maximum-paths eibgp 64
RP/0/RSP0/CPU0:router(config-bgp-af)# !
RP/0/RSP0/CPU0:router(config-bgp-af)# neighbor 1.0.0.2
RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 2
RP/0/RSP0/CPU0:router(config-bgp-nbr)# ebgp-multihop 255
RP/0/RSP0/CPU0:router(config-bgp-nbr)# dmz-link-bandwidth
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# multipath
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy BW1 in
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all out
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# !
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# neighbor 2.0.0.2
RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 2
RP/0/RSP0/CPU0:router(config-bgp-nbr)# ebgp-multihop 255
RP/0/RSP0/CPU0:router(config-bgp-nbr)# dmz-link-bandwidth
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# multipath
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy BW2 in
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all out

```




CHAPTER 7

Implementing BFD

Bidirectional forwarding detection (BFD) provides low-overhead, short-duration detection of failures in the path between adjacent forwarding engines. BFD allows a single mechanism to be used for failure detection over any media and at any protocol layer, with a wide range of detection times and overhead. The fast detection of failures provides immediate reaction to failure in the event of a failed link or neighbor.

- [Prerequisites for Implementing BFD, on page 297](#)
- [Restrictions for Implementing BFD, on page 298](#)
- [Information About BFD, on page 298](#)
- [BFD Hardware Offload for RSVP Tail-End, on page 316](#)
- [RFCs, on page 322](#)
- [Technical Assistance, on page 322](#)

Prerequisites for Implementing BFD

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

The following prerequisites are required to implement BFD:

- If enabling BFD on Multiprotocol Label Switching (MPLS), an installed composite PIE file including the MPLS package, or a composite-package image is required. For Border Gateway Protocol (BGP), Intermediate System-to-Intermediate System (IS-IS), Static, and Open Shortest Path First (OSPF), an installed Cisco IOS XR IP Unicast Routing Core Bundle image is required.
- Interior Gateway Protocol (IGP) is activated on the router if you are using IS-IS or OSPF.
- To enable BFD for a neighbor, the neighbor router must support BFD.
- To support BFD on bundle member links, be sure that the following requirements are met:
 - The routers on either end of the bundle are connected back-to-back without a Layer 2 switch in between.
 - For a BFD session to start, any one of the following configurations or states are present on the bundle member:
Link Aggregation Control Protocol (LACP) Distributing state is reached, –Or–
EtherChannel is configured.

Hot Standby and LACP Collecting state is reached.

Restrictions for Implementing BFD

These restrictions apply to BFD:

- Demand mode is not supported in Cisco IOS XR software.
- Echo latency detection and echo validation are not supported on bundle interfaces.
- Echo mode is not supported on BFD over bundle interfaces and on bundle VLANs, which is BFD over logical bundle (BLB).

Information About BFD

BFD Packet Intervals on Physical Interfaces

When BFD is running over physical interfaces, echo mode is used only if the configured interval is less than two seconds.

BFD sessions running over physical interfaces when echo mode is enabled send BFD control packets at a slow rate of every two seconds. There is no need to duplicate control packet failure detection at a fast rate because BFD echo packets are already being sent at fast rates and link failures will be detected when echo packets are not received within the echo failure detection time.

Control Packet Failure Detection In Asynchronous Mode

Control packet failure in asynchronous mode without echo is detected using the values of the minimum interval (`bfd minimum-interval` for non-bundle interfaces, and `bfd address-family ipv4 minimum-interval` for bundle interfaces) and multiplier (`bfd multiplier` for non-bundle interfaces, and `bfd address-family ipv4 multiplier` for bundle interfaces) commands.

For control packet failure detection, the local multiplier value is sent to the neighbor. A failure detection timer is started based on $(I \times M)$, where I is the negotiated interval, and M is the multiplier provided by the remote end.

Whenever a valid control packet is received from the neighbor, the failure detection timer is reset. If a valid control packet is not received from the neighbor within the time period $(I \times M)$, then the failure detection timer is triggered, and the neighbor is declared down.

Priority Settings for BFD Packets

For all interfaces under over-subscription, the internal priority needs to be assigned to remote BFD Echo packets, so that these BFD packets are not overwhelmed by other data packets. In addition, CoS values need to be set appropriately, so that in the event of an intermediate switch, the reply back of remote BFD Echo packets are protected from all other packets in the switch.

As configured CoS values in ethernet headers may not be retained in Echo messages, CoS values must be explicitly configured in the appropriate egress QoS service policy. CoS values for BFD packets attached to a traffic class can be set using the `set cos` command. For more information on configuring class-based unconditional packet marking, see “Configuring Modular QoS Packet Classification” in the .

BFD over Bundles IETF Mode Support on a Per Bundle Basis

BFD over Bundle (BoB) mode is a standard based fast failure detection of link aggregation (LAG) member links that is interoperable between different platforms. BoB support on a per bundle basis provides an option to choose IETF standard per bundle, without necessitating reloads or process restarts across various systems.

- IETF mode uses IANA assigned MAC.
- IETF BFD over Bundle sessions use destination UDP port: 6784.

Restrictions

These limitations apply for the BFD over Bundle Mode feature:

- You can use the `no bfd address-family ipv4 fast-detect` command to make BoB non-operational. You can also choose to configure a bundle to 'down' state by configuring shutdown under that particular bundle.
- For a bundle to accept the new BFD mode change, you must bring down and then recreate the existing BFD sessions.

BFD Dampening

Bidirectional Forwarding Detection (BFD) is a mechanism used by routing protocols to quickly realize and communicate the reachability failures to their neighbors. When BFD detects a reachability status change of a client, its neighbors are notified immediately. Sometimes it might be critical to minimize changes in routing tables so as not to impact convergence, in case of a micro failure. An unstable link that flaps excessively can cause other devices in the network to consume substantial processing resources, and that can cause routing protocols to lose synchronization with the state of the flapping link.

The BFD Dampening feature introduces a configurable exponential delay mechanism. This mechanism is designed to suppress the excessive effect of remote node reachability events flapping with BFD. The BFD Dampening feature allows the network operator to automatically dampen a given BFD session to prevent excessive notification to BFD clients, thus preventing unnecessary instability in the network. Dampening the notification to a BFD client suppresses BFD notification until the time the session under monitoring stops flapping and becomes stable.

Configuring the BFD Dampening feature, especially on a high-speed interface with routing clients, improves convergence time and stability throughout the network. BFD dampening can be applied to all types of BFD sessions, including IPv4/single-hop, Multiprotocol Label Switching-Transport Profile (MPLS-TP), and Pseudo Wire (PW) Virtual Circuit Connection Verification (VCCV).

BFD Session Dampening

You can configure the BFD Dampening feature at the BFD template level (single-hop template). Dampening is applied to all the sessions that use the BFD template. If you choose not to have a session to be dampened,

you should use a new BFD template without dampening for a new session. By default, the dampening functionality is not enabled on a template.

BFD Hardware Offload Support for IPv4

The Bidirectional Forwarding detection (BFD) Hardware Offload feature enables the offload of a BFD session to the network processing units of the line cards, in an IPv4 network. BFD hardware offload improves scale and reduces the overall network convergence time by sending rapid failure detection packets to the routing protocols for recalculating the routing table.

Restrictions

- This feature is not supported over MPLS LDP interface, VRRP interface, BVI interface and IRB interface.
- This feature is not supported over MPLS TE or RSVP tunnel.

Configuration Example

```

/* Configure BFD over Bundle(BOB) for hardware offload. */
Router# config
Router(config)# interface Bundle-Ether 1
Router(config-if)# bfd mode ietf
Router(config-if)# bfd address-family ipv4 multiplier 3
Router (config-if)# bfd address-family ipv4 destination 10.20.20.1
Router (config-if)# bfd address-family ipv4 fast-detect
Router(config-if)# bfd address-family ipv4 minimum-interval 1200
Router(config-if)# ipv4 address 10.20.20.2/30

/* Configure BFD with a static route. */
Router(config)# router static
Router(config-static)# address-family ipv4 unicast 10.1.1.0/24 10.6.0.2 bfd fast-detect
minimum-interval 1200 multiplier 4

/* Configure BFD with IS-IS. */
Router(config)# router isis 65444
Router(config-isis)# address-family ipv4 unicast
Router(config-isis)# exit
Router(config-isis)# interface HundredGige 0/3/0/1
Router(config-isis-if)# bfd minimum-interval 1200
Router(config-isis-if)# bfd multiplier 7
Router(config-isis-if)# bfd fast-detect ipv4
Router(config-isis-if)# address-family ipv4 unicast

/* Configure BFDv4 with OSPF. */
Router(config)# router ospf main
Router(config-ospfv3)# area 0
Router(config-ospfv3-ar)# interface HundredGige 0/0/0/1
Router(config-ospfv3-ar-if)# bfd multiplier 7
Router(config-ospfv3-ar-if)# bfd fast-detect
Router(config-ospfv3-ar-if)# bfd minimum-interval 1200

/* Configuring BFD over BGP. */
Router(config)# router bgp 120
Router(config-bgp)# neighbor 10.6.6.1
Router(config-bgp-nbr)# bfd fast-detect
Router(config-bgp-nbr)# bfd multiplier 7
Router(config-bgp-nbr)# bfd minimum-interval 1200

```

Verification

Use the **show bfd ipv4 session** command to verify the configuration:

```
Router# show bfd ipv4 session
Interface          Dest Addr          Local det time(int*mult)  State
                   Echo              Async   H/W                    NPU
-----
Hu0/0/0/22.93     10.20.20.1        0s(0s*0)                 12ms(4ms*3)  UP
                                                Yes   0/0/CPU0
```

BFD Hardware Offload Support for IPv6

The Bidirectional Forwarding detection (BFD) Hardware Offload feature enables the offload of a BFD session to the network processing units of the line cards, in an IPv6 network. BFD hardware offload feature improves scale and reduces the overall network convergence time by sending rapid failure detection packets to the routing protocols for recalculating the routing table.

Restrictions

- This feature is not supported over MPLS LDP interface, VRRP interface, BVI interface and IRB interface.
- This feature is not supported over MPLS TE or RSVP tunnel.
- BFD Dampening is not supported for BFD over IPv6.
- BFD over Bundle (BOB) over IPv6 is not supported with dynamically configured link-local address. It must be statically configured.

Configuration Example

```
/* Configure BFD over Bundle(BOB) for hardware offload. */
Router# config
Router(config)# interface Bundle-Ether 1
Router(config-if)# bfd mode ietf
Router(config-if)# bfd address-family ipv6 multiplier 3
Router (config-if)# bfd address-family ipv6 destination 10.20:20::1
Router (config-if)# bfd address-family ipv6 fast-detect
Router(config-if)# bfd address-family ipv6 minimum-interval 1200
Router(config-if)# ipv6 address 10:20:20::2/64

/* Configure BFD with a static route. */
Router(config)# router static
Router(config-static)# address-family ipv6 unicast 1011:17e4::1/128 ab11:15d2::2 bfd
fast-detect minimum-interval 1200 multiplier 3

/* Configure BFD with IS-IS. */
Router(config)# router isis 65444
Router(config-isis)# address-family ipv6 unicast
Router(config-isis)# exit
Router(config-isis)# interface HundredGige 0/3/0/1
Router(config-isis-if)# bfd minimum-interval 1200
Router(config-isis-if)# bfd multiplier 7
Router(config-isis-if)# bfd fast-detect ipv6
Router(config-isis-if)# address-family ipv6 unicast

/* Configure BFDv6 with OSPFv3. */
Router(config)# router ospfv3 main
Router(config-ospfv3)# area 0
Router(config-ospfv3-ar)# interface HundredGige 0/0/0/1
```

```

Router(config-ospfv3-ar-if)# bfd multiplier 7
Router(config-ospfv3-ar-if)# bfd fast-detect
Router(config-ospfv3-ar-if)# bfd minimum-interval 1200

/* Configuring BFD over BGP. */
Router(config)# router bgp 120
Router(config-bgp)# neighbor 2001:DB8:1::1
Router(config-bgp-nbr)# bfd fast-detect
Router(config-bgp-nbr)# bfd multiplier 7
Router(config-bgp-nbr)# bfd minimum-interval 1200

```

Verification

Use the **show bfd ipv6 session** command to verify the configuration:

```

Router# show bfd ipv6 session
Interface          Dest Addr
-----
H/W                NPU                Echo                Async                State
-----
BE7.2              fe80::28a:96ff:fed6:9cdb
Yes                0/0/CPU0           0s(0s*0)            900ms(300ms*3)     UP
BE7.4              fe80::28a:96ff:fed6:9cdb
Yes                0/0/CPU0           0s(0s*0)            900ms(300ms*3)     UP

```

IPv4 Multihop BFD

IPv4 Multihop BFD is a BFD session between two addresses between two nodes. An example of this feature is a BFD session between PE and CE loopback addresses or BFD sessions between routers that are several TTL hops away. The applications that support IPv4 Multihop BFD are external and internal BGP. IPv4 Multihop BFD feature supports BFD on arbitrary paths, which can span multiple network hops.

The IPv4 Multihop BFD feature provides sub-second forwarding failure detection for a destination more than one hop, and up to 255 hops, away. The **bfd multihop ttl-drop-threshold** command can be used to drop BFD packets coming from neighbors exceeding a certain number of hops.

Configure IPv4 Multihop BFD

This section describes how you can configure IPv4 Multihop BFD feature.

```

Router# configure
Router(config)# bfd
Router(config)# multihop ttl-drop-threshold 225
Router(config)# multipath include location 0/7/CPU0
Router(config)# router bgp 100
Router(config-bgp)# neighbor 209.165.200.225
Router(config-bgp-nbr)# remote-as 2000
Router(config-bgp-nbr)# update-source loopback 1
Router(config-bgp-nbr)# bfd fast-detect
Router(config-bgp-nbr)# bfd multiplier 3
Router(config-bgp-nbr)# bfd minimum-interval 1200
Router(config-bgp-nbr-af)# route-policy pass-all in
Router(config-bgp-nbr-af)# route-policy pass-all out
Router(config-bgp-nbr-af)# commit

```

Running Configuration

```

bfd
 multihop ttl-drop-threshold 225

```

```

multipath include location 0/7/CPU0
router bgp 100
 neighbor 209.165.200.225
   remote-as 2000
   update-source loopback 1
   bfd fast-detect
   bfd multiplier 3
   bfd minimum-interval 1200
     route-policy PASS-ALL in
     route-policy PASS-ALL out
!
!

```

Verification

The show outputs given in the following section display the details of the configuration of the IPv4 Multihop BFD feature, and the status of their configuration.

```

Router# show tech-support bfdhwoff
harddisk:
Tue Mar 20 11:20:29.214 PDT
++ Show tech start time: 2018-Mar-20.112029.PDT ++
Tue Mar 20 11:20:30 PDT 2018 Waiting for gathering to complete .....
Tue Mar 20 11:22:37 PDT 2018 Compressing show tech output Show tech output available at
0/RP0/CPU0 :
/harddisk:/showtech-bfd-hwoff-platform-2018-Mar-20.112029.PDT.tgz
++ Show tech end time: 2018-Mar-20.112237.PDT ++

```

Configure BFD

Configure BFD Under a Dynamic Routing Protocol or Use a Static Route

To establish a BFD neighbor, complete at least one of the following procedures to configure BFD under a dynamic routing protocol or to use a static route:

Enabling BFD on a BGP Neighbor

BFD can be enabled per neighbor, or per interface. This task describes how to enable BFD for BGP on a neighbor router.

Step 1 **configure**

Example:

```
RP/0/# configure
Enters mode.
```

Step 2 **router bgp** *autonomous-system-number*

Example:

```
RP/0/RP0/CPU0:router(config)# router bgp 120
Enters BGP configuration mode, allowing you to configure the BGP routing process.
```

Step 3 **neighbor** *ip-address*

Example:

```
RP/0/RP0/CPU0:router(config-bgp)# neighbor 172.168.40.24
```

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer. This example configures the IP address 172.168.40.24 as a BGP peer.

Step 4 `remote-as` *autonomous-system-number***Example:**

```
RP/0/RP0/CPU0:router(config-bgp-nbr)# remote-as 2002
```

Creates a neighbor and assigns it a remote autonomous system.

This example configures the remote autonomous system to be 2002.

Step 5 `bfd fast-detect`**Example:**

```
RP/0/RP0/CPU0:router(config-bgp-nbr)# bfd fast-detect
```

Enables BFD between the local networking devices and the neighbor whose IP address you configured to be a BGP peer in Step 3.

In the example in Step 3, the IP address 172.168.40.24 was set up as the BGP peer. In this example, BFD is enabled between the local networking devices and the neighbor 172.168.40.24.

Step 6 `bfd minimum-interval` *milliseconds***Example:**

```
RP/0/RP0/CPU0:router(config-bgp-nbr)# bfd minimum-interval 1200
```

Sets the BFD minimum interval. Range is 3-1200 milliseconds.

Step 7 `bfd multiplier` *multiplier***Example:**

```
RP/0/RP0/CPU0:router(config-bgp-nbr)# bfd multiplier 7
```

Sets the BFD multiplier. This is optional, the minimum is 3 and by default the multiplier will be 3 for all protocols

Step 8 Use the `commit` or `end` command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Enabling BFD for OSPF on an Interface

Perform the following steps to configure BFD for Open Shortest Path First (OSPF) on an interface. The steps in the procedure are common to the steps for configuring BFD on IS-IS; only the command mode differs.



Note BFD per interface configuration is supported for OSPF and IS-IS only.

```
Router# configure

/* Enter OSPF configuration mode to configure the OSPF routing process. */
Router(config)# router ospf 0

/* Set the BFD minimum interval. The range is from 3 to 1200 milliseconds. */
Router(config-ospf)# bfd minimum-interval 1200

/* Set the BFD multiplier. */
Router(config-ospf)# bfd multiplier 7

/* Configure an Open Shortest Path First (OSPF) area. */
Router(config-ospf)# area 0

/* Enter interface configuration mode. */
Router(config-ospf-ar)# interface HundredGige 0/3/0/1

/* Enable BFD to detect failures in the path between adjacent forwarding engines. */
Router(config-ospf-ar-if)# bfd fast-detect
```

Running Configuration

```
configure
  router ospf 0
  bfd minimum-interval 1200
  bfd multiplier 7
  area 0
  interface HundredGige 0/3/0/1
  bfd fast-detect
```

Verification

Verify that BFD is enabled on the appropriate interface.

```
Router(config-ospf-ar-if)# show run router ospf

router ospf 0
bfd minimum-interval 1200
bfd multiplier 7
area 0
interface HundredGige 0/3/0/1
bfd fast-detect
```

Enabling BFD on a Static Route

The following procedure describes how to enable BFD on a static route.

```
Router(config)# configure

/*Enter static route configuration mode, and configure static routing. */
```

```

Router(config)# router static

/*Enter address family configuration mode. */
Router(config-static)# address-family ipv4 unicast 192.168.2.2/32

/*Specify an unicast destination address and next-hop IPv4 address.
Enable BFD fast-detection on the specified IPv4 unicast destination address */
Router(config-static)# 192.168.2.2 192.168.6.2 bfd fast-detect minimum-interval 1200
multiplier 3

```

Running Configuration

```

router static
 address-family ipv4 unicast
   192.168.2.2 192.168.6.2 bfd fast-detect minimum-interval 1200 multiplier 3
!
!

```

Specifying the BFD Destination Address on a Bundle

To specify the BFD destination address on a bundle, complete these steps:

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 **interface Bundle-Ether *bundle-id***

Example:

```
RP/0/RP0/CPU0:router(config)# interface Bundle-Ether 1
```

Enters interface configuration mode for the specified bundle ID.

Step 3 **bfd address-family ipv4 destination *ip-address***

Example:

```
RP/0/RP0/CPU0:router(config-if)# bfd address-family ipv4 destination 10.20.20.1
```

Specifies the primary IPv4 address assigned to the bundle interface on a connected remote system, where *ip-address* is the 32-bit IP address in dotted-decimal format (A.B.C.D).

Step 4 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.

- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Enabling BFD Sessions on Bundle Members

To enable BFD sessions on bundle member links, complete these steps:

SUMMARY STEPS

1. **configure**
2. **interface Bundle-Ether** *bundle-id*
3. **bfd address-family ipv4 fast-detect**
4. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 **interface Bundle-Ether** *bundle-id*

Example:

```
Router(config)# interface Bundle-Ether 1
```

Enters interface configuration mode for the specified bundle ID.

Step 3 **bfd address-family ipv4 fast-detect**

Example:

```
Router(config-if)# bfd address-family ipv4 fast-detect
```

Enables IPv4 BFD sessions on bundle member links.

Step 4 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
 - **No** —Exits the configuration session without committing the configuration changes.
 - **Cancel** —Remains in the configuration session, without committing the configuration changes.
-

Configuring the Minimum Thresholds for Maintaining an Active Bundle

The bundle manager uses two configurable minimum thresholds to determine whether a bundle can be brought up or remain up, or is down, based on the state of its member links.

- Minimum active number of links
- Minimum active bandwidth available

Whenever the state of a member changes, the bundle manager determines whether the number of active members or available bandwidth is less than the minimum. If so, then the bundle is placed, or remains, in DOWN state. Once the number of active links or available bandwidth reaches one of the minimum thresholds, then the bundle returns to the UP state.

To configure minimum bundle thresholds, complete these steps:

Step 1 **configure**

Step 2 **interface** **Bundle-Ether** *bundle-id*

Example:

```
RP/0/RP0/CPU0:router(config)# interface Bundle-Ether 1
```

Enters interface configuration mode for the specified bundle ID.

Step 3 **bundle minimum-active bandwidth** *kbps*

Example:

```
RP/0/RP0/CPU0:router(config-if)# bundle minimum-active bandwidth 580000
```

Sets the minimum amount of bandwidth required before a bundle can be brought up or remain up. The range is from 1 through a number that varies depending on the platform and the bundle type.

Step 4 **bundle minimum-active links** *links*

Example:

```
RP/0/RP0/CPU0:router(config-if)# bundle minimum-active links 2
```

Sets the number of active links required before a bundle can be brought up or remain up. The range is from 1 to 32.

Note When BFD is started on a bundle that is already active, the BFD state of the bundle is declared when the BFD state of all the existing active members is known.

Step 5 **commit**

Configuring BFD Packet Transmission Intervals and Failure Detection Times on a Bundle

BFD asynchronous packet intervals and failure detection times for BFD sessions on bundle member links are configured using a combination of the **bfd address-family ipv4 minimum-interval** and **bfd address-family ipv4 multiplier** interface configuration commands on a bundle.

The BFD control packet interval is configured directly using the **bfd address-family ipv4 minimum-interval** command. The failure detection times are determined by a combination of the interval and multiplier values in these commands.

To configure the minimum transmission interval and failure detection times for BFD asynchronous mode control packets on bundle member links, complete these steps:

Step 1 **configure**

Step 2 **interface Bundle-Ether** *bundle-id*

Example:

```
RP/0/RP0/CPU0:router(config)# interface Bundle-Ether 1
```

Enters interface configuration mode for the specified bundle ID.

Step 3 **bfd address-family ipv4 minimum-interval** *milliseconds*

Example:

```
RP/0/RP0/CPU0:router(config-if)#bfd address-family ipv4 minimum-interval 1200
```

Note Specifies the minimum interval, in milliseconds, for asynchronous mode control packets on IPv4 BFD sessions on bundle member links. The range is from 3 to 1200.

Step 4 **bfd address-family ipv4 multiplier** *multiplier*

Example:

```
RP/0/RP0/CPU0:router(config-if)#bfd address-family ipv4 multiplier 30
```

Specifies a number that is used as a multiplier with the minimum interval to determine BFD control packet failure detection times and transmission intervals for IPv4 BFD sessions on bundle member links.

Note Although the command allows you to configure a minimum of 2, the supported minimum is 3.

Step 5 **commit**

Configure BFD over Bundles IETF Mode Support on a Per Bundle Basis

To configure BFD over Bundles IETF mode support on a per bundle basis use these steps:

Step 1 **configure**

Step 2 **interface Bundle-Ether** *bundle-id*

Example:

```
RP/0/RP0/CPU0:router(config)# interface Bundle-Ether 1
```

Enters interface configuration mode for the specified bundle ID.

Step 3 **bfd mode ietf**

Example:

```
RP/0/RP0/CPU0:router(config-if)# bfd mode ietf
```

Enables IETF mode for BFD over bundle for the specified bundle.

Step 4 **bfd address-family ipv4 fast-detect**

Example:

```
RP/0/RP0/CPU0:router(config-if)# bfd address-family ipv4 fast-detect
```

Enables IPv4 BFD sessions on the specified bundle.

Step 5 `commit`

Step 6 `show bundle bundle-ether bundle-id`

Displays the selected bundle mode.

Enabling Echo Mode to Test the Forwarding Path to a BFD Peer

BFD echo mode is enabled by default for IPv4 on other physical interfaces whose minimum interval is less than three seconds.

If you have configured a BFD minimum interval greater than three seconds on a physical interface using the **bfd minimum-interval** command, then you will need to change the interval to be less than three seconds to support and enable echo mode. This does not apply to bundle member links, which always support echo mode.

Overriding the Default Echo Packet Source Address

If you do not specify an echo packet source address, then BFD uses the IP address of the output interface as the default source address for an echo packet.

You can use the **echo ipv4 source** command in BFD or interface BFD configuration mode to specify the IP address that you want to use as the echo packet source address.

You can override the default IP source address for echo packets for BFD on the entire router, or for a particular interface.

Specifying the Echo Packet Source Address Globally for BFD

To specify the echo packet source IP address globally for BFD on the router, complete the following steps:

Step 1 `configure`

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 `bfd`

Example:

```
Router(config)# bfd
```

Enters BFD configuration mode.

Step 3 `echo ipv4 source ip-address`

Example:

```
Router(config-bfd)# echo ipv4 source 10.10.10.1
```

Specifies an IPv4 address to be used as the source address in BFD echo packets, where *ip-address* is the 32-bit IP address in dotted-decimal format (A.B.C.D).

Step 4 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Specifying the Echo Packet Source Address on an Individual Interface

To specify the echo packet source IP address on an individual BFD interface, complete the following steps:

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 **bfd**

Example:

```
Router(config)# bfd
```

Enters BFD configuration mode.

Step 3 **interface** type interface-path-id

Example:

```
Router(config-bfd)# interface HundredGige 0/1/5/0
```

Enters BFD interface configuration mode for a specific interface. In BFD interface configuration mode, you can specify an IPv4 address on an individual interface.

Step 4 **echo ipv4 source** *ip-address*

Example:

```
Router(config-bfd)# echo ipv4 source 10.10.10.1
```

Specifies an IPv4 address to be used as the source address in BFD echo packets, where *ip-address* is the 32-bit IP address in dotted-decimal format (A.B.C.D).

Step 5 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Disabling Echo Mode

BFD does not support asynchronous operation in echo mode in certain environments.

You can disable echo mode for BFD on the entire router, or for a particular interface.

Disabling Echo Mode on a Router

To disable echo mode globally on the router complete the following steps:

DETAILED STEPS

Step 1 **configure**

Step 2 **bfd**

Example:

```
Router(config)# bfd
```

Enters BFD configuration mode.

Step 3 **echo disable**

Example:

```
Router(config-bfd)# echo disable
```

Disables echo mode on the router.

Step 4 **commit**

Disabling Echo Mode on an Individual Interface

The following procedures describe how to disable echo mode on an interface.

Step 1 **configure**

Example:

```
RP/0/# configure
```

Enters mode.

Step 2 **bfd**

Example:

```
Router(config)# bfd
```

Enters BFD configuration mode.

Step 3 **interface** *type interface-path-id***Example:**

```
Router(config-bfd)# interface HundredGige 0/1/5/0
```

Enters BFD interface configuration mode for a specific interface. In BFD interface configuration mode, you can disable echo mode on an individual interface.

Step 4 **echo disable****Example:**

```
Router(config-bfd-if)# echo disable
```

Disables echo mode on the specified individual interface.

Step 5 **commit**

Minimizing BFD Session Flapping Using BFD Dampening

To configure BFD dampening to control BFD session flapping, complete the following steps.

Step 1 **configure****Example:**

```
RP/0/# configure
```

Enters mode.

Step 2 **bfd****Example:**

```
Router(config)# bfd
```

Enters BFD configuration mode.

Step 3 **dampening** [**bundle-member**] {**initial-wait** | **maximum-wait** | **secondary-wait**} *milliseconds***Example:**

```
Router(config-bfd)# dampening initial-wait 30000
```

Specifies delays in milliseconds for BFD session startup to control flapping.

The value for **maximum-wait** should be greater than the value for **initial-wait**.

The dampening values can be defined for bundle member interfaces and for the non-bundle interfaces.

Step 4 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Clear and Display BFD Counters

The following procedure describes how to display and clear BFD packet counters. You can clear packet counters for BFD sessions that are hosted on a specific node or on a specific interface.

```
Router# show bfd counters all packet location 0/3/cpu0
Router# clear bfd counters all packet location 0/3/cpu0
Router# show bfd counters all packet location 0/3/cpu0
```

BFD IPv6 in Bundle Manager Domain

A configuration to enable or disable BFD to run over a bundle interface can be in the bundle manager domain. The bundle manager can apply these configuration changes, and based on the configuration changes, request the BFD server to enable or disable BFD on certain bundle interfaces and a member links related to those bundle interfaces.

BFD Over BGP: Example

The following example shows how to configure BFD between autonomous system 1200 and neighbor 192.168.70.24:

```
Router#configure
Router#(config)#router bgp 1200
Router#(config-bgp)#bfd multiplier 2
Router#(config-bgp)#bfd minimum-interval 1200
Router#(config-bgp)#neighbor 192.168.70.24
Router#(config-bgp-nbr)#remote-as 2
Router#(config-bgp-nbr)#bfd fast-detect
Router#(config-bgp-nbr)#commit
Router#(config-bgp-nbr)#end
Router##show run router bgp
```

BFD Over OSPF: Example

The following example shows how to enable BFD for OSPF on a HundredGigE interface:

```
Router#configure
Router#(config)#router ospf 0
Router#(config-ospf)#area 0
Router#(config-ospf-ar)#interface HundredGige 0/3/0/1
Router#(config-ospf-ar-if)#bfd fast-detect
Router#(config-ospf-ar-if)#commit
```

```
Router#(config-ospf-ar-if)#end

Router#show run router ospf

router ospf 0
area 0
interface HundredGige 0/3/0/1
bfd fast-detect
```

BFD Over Static Routes: Example

The following example shows how to enable BFD on an IPv4 static route. In this example, BFD sessions are established with the next-hop 10.3.3.3 when it becomes reachable.

```
Router#configure
Router(config)#router static
Router(config-static)#address-family ipv4 unicast
Router(config-static)#10.2.2.0/24 10.3.3.3 bfd fast-detect
Router(config-static)#end
```

BFD Echo Mode Disable: Examples

The following example shows how to disable echo mode on a router:

```
Router#configure
Router(config)#bfd
Router#r(config-bfd)#echo disable
```

The following example shows how to disable echo mode on an interface:

```
Router##configure
Router#(config)#bfd
Router#(config-bfd)#interface HundredGige 0/1/0/0
Router#(config-bfd-if)#echo disable
```

Echo Packet Source Address: Examples

The following example shows how to specify the IP address 10.10.10.1 as the source address for BFD echo packets for all BFD sessions on the router:

```
Router#configure
Router(config)#bfd
Router(config-bfd)#echo ipv4 source 10.10.10.1
```

The following example shows how to specify the IP address 10.10.10.1 as the source address for BFD echo packets on an individual HundredGige Ethernet interface:

```
Router#configure
Router(config)#bfd
Router(config-bfd)#interface HundredGige 0/1/0/0
Router(config-bfd-if)#echo ipv4 source 10.10.10.1
```

BFD Dampening: Examples

The following example shows how to configure an initial and maximum delay for BFD session startup on BFD bundle members:

```
Router#configure
Router(config)#bfd
Router(config-bfd)#dampening bundle-member initial-wait 8000
Router(config-bfd)#dampening bundle-member maximum-wait 15000
```

The following example shows how to change the default initial-wait for BFD on a non-bundle interface:

```
Router#configure
Router(config)#bfd
Router(config-bfd)#dampening initial-wait 30000
Router(config-bfd)#dampening maximum-wait 35000
```

BFD Hardware Offload for RSVP Tail-End

Table 10: Feature History Table

You can use Bidirectional Forwarding Detection (BFD) to detect Multiprotocol Label Switching (MPLS) Label Switched Path (LSP) data plane failures. An LSP ping request is used for detecting MPLS data plane failures and also for verifying the data plane against the control plane. BFD cannot be used for verifying the data plane against the control plane. However, the control plane processing required for BFD control packets is relatively smaller than the processing required for LSP ping messages. Hence, BFD can be deployed for faster detection of data plane failure (for example, traffic black-holing) for a large number of LSPs.

This feature improves the scale and reduces the overall network convergence time by sending rapid failure detection packets to the routing protocols for recalculating the routing table.

BFD over MPLS Traffic Engineering LSPs

Bidirectional Forwarding Detection (BFD) over MPLS Traffic Engineering Label Switched Paths (LSPs) feature in Cisco IOS XR Software detects MPLS Label Switched Path LSP data plane failures. Since the control plane processing required for BFD control packets is relatively smaller than the processing required for LSP Ping messages, BFD can be deployed for faster detection of data plane failure for a large number of LSPs.

The BFD over MPLS TE LSPs implementation in Cisco IOS XR Software is based on *RFC 5884: Bidirectional Forwarding Detection (BFD) for MPLS Label Switched Paths (LSPs)*. LSP Ping is an existing mechanism for detecting MPLS data plane failures and for verifying the MPLS LSP data plane against the control plane. BFD can be used for detecting MPLS data plane failures, but not for verifying the MPLS LSP data plane against the control plane. A combination of LSP Ping and BFD provides faster data plane failure detection on a large number of LSPs.

The BFD over MPLS Tail-End LSPs is used for networks that have deployed MPLS as the multi service transport and that use BFD as fast failure detection mechanism to enhance network reliability and up time by using BFD as fast failure detection traffic black holing.

BFD process interacts with the Tail-End and LSPV processes to support BFD over TE LSP feature. MPLS Tail-End automatically establishes and maintains the LSPs across the MPLS network by using the Resource Reservation Protocol (RSVP).

To know how MPLS works with RSVP Tail-End, refer to the *MPLS Configuration guide for Cisco 8000 Routers*.

BFD over MPLS Tail-End LSPs support:

- BFD async mode (BFD echo mode is not supported)
- IPv4 only, since MPLS core is IPv4
- BFD packets that carry IP DSCP 6 (Internet Control)
- Use of BFD for TE tunnel bring up, re-optimization, and path protection (Standby and FRR)
- Fastest detection time (3 ms x 3 = 9 ms)
- Optional Periodic LSP ping verification after BFD session is up
- Dampening to hold-down BFD failed path-option

There are two ways in which the BFD packets from tail-end to head-end will be used:

- BFD packets from tail-end to head-end will be IP routed
- BFD packets from tail-end to head-end will be Label Switched if MPLS LDP is available in Core with label path from tail-end to head-end.

Configuring BFD over MPLS Traffic Engineering LSPs

.

Enabling BFD Parameters for BFD over TE Tunnels

BFD for TE tunnel is enabled at the head-end by configuring BFD parameters under the tunnel. When BFD is enabled on the already up tunnel, TE waits for the bringup timeout before bringing down the tunnel. BFD is disabled on TE tunnels by default. Perform these tasks to configure BFD parameters and enable BFD over TE Tunnels.



Note BFD paces the creation of BFD sessions by limiting LSP ping messages to be under 50 PPS to avoid variations in CPU usage.

1. Enter global configuration mode.

```
Router#config
```

2. Configure MPLS OAM.

```
Router (config) # mpls oam
```

3. Configure MPLS Traffic Engineering (MPLS TE) tunnel interface and enter into MPLS TE tunnel interface configuration mode.

```
Router (config) #interface tunnel-te 65535
```

4. Enable BFD fast detection.

```
Router(config-if)#bfd fast-detect
```

5. Configure hello interval in milliseconds.

```
Router(config-if)#bfd minimum-interval 500
```



Note Hello interval range is 3 to 1000 milliseconds. Default hello interval is 100 milliseconds.

6. Configure BFD multiplier detection.

```
Router(config-if)#bfd multiplier 5
```



Note BFD multiplier range is 3 to 10. Default BFD multiplier is 3.

7. Commit the changes.

```
Router(config-if)#commit
```

Configuring BFD Bring up Timeout

Perform these steps to configure BFD bring up timeout interval.

1. Enter global configuration mode.

```
Router#config
```

2. Configure MPLS Traffic Engineering (MPLS TE) tunnel interface and enter into MPLS TE tunnel interface configuration mode.

```
Router(config)#interface tunnel-te 65535
```

3. Enable the time interval (in seconds) to wait for the BFD session to come up.

```
Router(config-if)#bfd bringup-timeout 2400
```



Note The timeout range is 6 to 3600 seconds. Default bring up timeout interval is 60 seconds.

4. Commit the changes.

```
Router(config-if)#commit
```

Configuring BFD Dampening for TE Tunnels

When BFD session fails to come up, TE exponentially backs off using the failed path-option to avoid signaling churn in the network. Perform these steps to configure dampening intervals to bring the TE tunnel up.

1. Enter global configuration mode.

```
Router# configure
```

2. Configure MPLS Traffic Engineering (MPLS TE) tunnel interface and enter into MPLS TE tunnel interface configuration mode.

```
Router(config)#interface tunnel-te 65535
```

3. Configure the initial delay interval before bringing up the tunnel.

```
Router(config-if)#bfd dampening initial-wait 360000
```



Note The initial-wait bring up delay time interval range is 1 to 518400000 milliseconds. Default initial-wait interval is 16000 milliseconds.

4. Configure the maximum delay interval before bringing up the tunnel.

```
Router(config-if)#bfd dampening maximum-wait 700000
```



Note The maximum-wait bring up delay time interval range is 1 to 518400000 milliseconds. Default initial-wait interval is 600000 milliseconds.

5. Configure the secondary delay interval before bringing up the tunnel.

```
Router(config-if)#bfd dampening secondary-wait 30000
```



Note The secondary-wait bring up delay time interval range is 1 to 518400000 milliseconds. Default secondary-wait interval is 20000 milliseconds.

6. Commit the changes.

```
Router(config-if)#commit
```

Configuring Periodic LSP Ping Requests

Perform this task to configure sending periodic LSP ping requests with BFD TLV, after BFD session comes up.

1. Enter global configuration mode.

```
Router# configure
```

2. Configure MPLS Traffic Engineering (MPLS TE) tunnel interface and enter into MPLS TE tunnel interface configuration mode.

```
Router(config)#interface tunnel-te 65535
```

3. Set periodic interval for LSP ping requests in seconds.

```
Router(config-if)#bfd lsp-ping interval 300
```



Note The interval range is 60 to 3600 seconds. Default interval is 120 seconds.

4. Commit the changes.

```
Router(config-if)#commit
```

Configuring BFD at the Tail-End

Use the tail-end global configuration commands to set the BFD minimum-interval and BFD multiplier parameters for all BFD over LSP sessions. The ranges and default values are the same as the BFD head-end configuration values. BFD will take the maximum value set between head-end minimum interval and tail-end minimum interval. Perform these tasks to configure BFD at the tail-end.

1. Enter global configuration mode.

```
Router# configure
```

2. Configure MPLS OAM.

```
Router(config)# mpls oam
```

3. Configure hello interval in milliseconds.

```
Router(config)#mpls traffic-eng bfd lsp tail minimum-interval 500
```



Note Hello interval range is 3 to 1000 milliseconds. Default hello interval is 100 milliseconds.

4. Configure BFD multiplier detection.

```
Router(config)#mpls traffic-eng bfd lsp tail multiplier 5
```



Note BFD multiplier detect range is 3 to 10. Default BFD multiplier is 3.

5. Commit the changes.

```
Router(config-if)#commit
```

Configuring BFD over LSP Sessions on Line Cards

BFD over LSP sessions, both head-end and tail-end, are hosted on line cards with following configuration enabled.



Note For fixed box and centralized platforms, there are no line cards. Datapath is running on routing processors (RPs) which is where BFD sessions need to be created. On fixed box, the configuration must include the RPs instead of LCs. On centralized platforms, you cannot use RP in the config even though BFD sessions will be running on the RPs. You must include one of the MPAs instead of LC in the configuration.

1. Enter global configuration mode.

```
Router# configure
```

2. Enter BFD configuration mode.

```
Router(config)# bfd
```


- Configure BFD multiple path on specific line card.

```
Router(config-bfd)# multipath include location 0/1/CPU0
```

- Commit the changes.

```
Router(config-if)#commit
```

BFD over MPLS TE Tunnel Tail-End Configuration

You can use the `mpls traffic-eng bfd lsp tail minimum-interval` command to configure the tail-end at a minimum interval of 3 milli seconds.

Configuration

```
Router#config
Router(config)#mpls traffic-eng bfd lsp tail minimum-interval 3
Router(config)#commit
```

Running Configuration

```
mpls traffic-eng bfd lsp tail minimum-interval 3
!
```

Verification

Use the `show bfd session` command to verify the configuration on tail-end.

```
Router#show bfd session
```

Src Addr	Dest Addr	VRF Name	Type	Specific Data	State
H/W	NPU	Local det time(int*mult) Echo	Async		
1.1.1.1	2.2.2.2	default	TT32768 (LSP:2)	1500ms (500ms*3)	UP

Use the `show bfd label session` to verify the configuration on head-end.

```
Router#show bfd label session
```

Interface	Label	Local det time(int*mult) Echo	State
H/W	NPU	Async	
tt1 (LSP:103)	24001	150ms (50ms*3)	UP
Yes	0/1/CPU0		
tt2 (LSP:102)	24002	150ms (50ms*3)	UP
Yes	0/1/CPU0		
tt3 (LSP:101)	24004	150ms (50ms*3)	UP
Yes	0/1/CPU0		
tt4 (LSP:103)	24005	150ms (50ms*3)	UP
Yes	0/1/CPU0		
tt5 (LSP:104)	24006	150ms (50ms*3)	UP
Yes	0/1/CPU0		

Configuration Examples for Configuring BFD

BFD over MPLS TE LSPs Examples

These examples explain how to configure BFD over MPLS TE LSPs.

BFD Over MPLS TE Tunnel Head-End Configuration: Example

This example shows how to configure BFD over MPLS TE Tunnel at head-end.

```
Router# bfd multipath include loc 0/1/CPU0
mpls oam
interface tunnel-te 1
  bfd
    minimum-interval 500
    fast-detect
    multiplier 5
    bringup-timeout 60
    lsp-ping disable
    dampening initial-wait (default 16000 ms)
    dampening maximum-wait (default 600000 ms)
    dampening secondary-wait (default 20000 ms)
    logging events bfd-status
```

BFD Over MPLS TE Tunnel Tail-End Configuration: Example

This example shows how to configure BFD over MPLS TE Tunnels at tail-end.

```
Router# bfd multipath include loc 0/1/CPU0
mpls oam
mpls traffic-eng bfd lsp tail multiplier 3
mpls traffic-eng bfd lsp tail minimum-interval 500
```

RFCs

RFCs	Title
rfc5880_bfd_base	<i>Bidirectional Forwarding Detection</i> , June 2010
rfc5881_bfd_ipv4_ipv6	<i>BFD for IPv4 and IPv6 (Single Hop)</i> , June 2010

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport



CHAPTER 8

Implementing Fast Reroute Loop-Free Alternate

Fast Reroute Loop-Free Alternate feature enables you to tunnel a packet around a failed link to a remote loop-free alternate that is more than one hop away.

- [Prerequisites for IPv4/IPv6 Loop-Free Alternate Fast Reroute, on page 323](#)
- [Restrictions for Loop-Free Alternate Fast Reroute, on page 323](#)
- [IS-IS and IP FRR, on page 324](#)
- [Repair Paths, on page 324](#)
- [LFA Overview, on page 324](#)
- [LFA Calculation, on page 325](#)
- [Interaction Between RIB and Routing Protocols, on page 325](#)
- [Fast Reroute with Remote Loop-Free Alternate, on page 326](#)
- [Configuration , on page 327](#)
- [Configuring IPv4 or IPv6 Loop-Free Alternate Fast Reroute Support: Example, on page 330](#)

Prerequisites for IPv4/IPv6 Loop-Free Alternate Fast Reroute

Loop-Free Alternate (LFA) Fast Reroute (FRR) can protect paths that are reachable through an interface only if the interface is a point-to-point interface.

Restrictions for Loop-Free Alternate Fast Reroute

- The remote LFA backup path for MPLS traffic can be setup only using LDP.
- LFA calculations are restricted to interfaces or links belonging to the same level or area. Hence, excluding all neighbors on the same LAN when computing the backup LFA can result in repairs being unavailable in a subset of topologies.
- Only physical interfaces are protected. Subinterfaces, tunnels, and virtual interfaces are not protected.
- IPv4 and IPv6 LFA FRR not supported over TE tunnel.

IS-IS and IP FRR

When a local link fails in a network, IS-IS recomputes new primary next-hop routes for all affected prefixes. These prefixes are updated in the RIB and the Forwarding Information Base (FIB). Until the primary prefixes are updated in the forwarding plane, traffic directed towards the affected prefixes are discarded. This process can take hundreds of milliseconds.

In IP FRR, IS-IS computes LFA next-hop routes for the forwarding plane to use in case of primary path failures. LFA is computed per prefix.

When there are multiple LFAs for a given primary path, IS-IS uses a tiebreaking rule to pick a single LFA for a primary path. In case of a primary path with multiple LFA paths, prefixes are distributed equally among LFA paths.

Repair Paths

Repair paths forward traffic during a routing transition. When a link or a router fails, due to the loss of a physical layer signal, initially, only the neighboring routers are aware of the failure. All other routers in the network are unaware of the nature and location of this failure until information about this failure is propagated through a routing protocol, which may take several hundred milliseconds. It is, therefore, necessary to arrange for packets affected by the network failure to be steered to their destinations.

A router adjacent to the failed link employs a set of repair paths for packets that would have used the failed link. These repair paths are used from the time the router detects the failure until the routing transition is complete. By the time the routing transition is complete, all routers in the network revise their forwarding data and the failed link is eliminated from the routing computation.

Repair paths are precomputed in anticipation of failures so that they can be activated the moment a failure is detected.

The LFA FRR feature uses the following repair paths:

- Equal Cost Multipath (ECMP) uses a link as a member of an equal cost path-split set for a destination. The other members of the set can provide an alternative path when the link fails.
- LFA is a next-hop route that delivers a packet to its destination without looping back. Downstream paths are a subset of LFAs.

LFA Overview

LFA is a node other than the primary neighbor. Traffic is redirected to an LFA after a network failure. An LFA makes the forwarding decision without any knowledge of the failure.

An LFA must neither use a failed element nor use a protecting node to forward traffic. An LFA must not cause loops. By default, LFA is enabled on all supported interfaces as long as the interface can be used as a primary path.

Advantages of using per-prefix LFAs are as follows:

- The repair path forwards traffic during transition when the primary path link is down.

- All destinations having a per-prefix LFA are protected. This leaves only a subset (a node at the far side of the failure) unprotected.

LFA Calculation

The general algorithms to compute per-prefix LFAs can be found in RFC 5286. IS-IS implements RFC 5286 with a small change to reduce memory usage. Instead of performing a Shortest Path First (SPF) calculation for all neighbors before examining prefixes for protection, IS-IS examines prefixes after SPF calculation is performed for each neighbor. Because IS-IS examines prefixes after SPF calculation is performed, IS-IS retains the best repair path after SPF calculation is performed for each neighbor. IS-IS does not have to save SPF results for all neighbors.

Interaction Between RIB and Routing Protocols

A routing protocol computes repair paths for prefixes by implementing tiebreaking algorithms. The end result of the computation is a set of prefixes with primary paths, where some primary paths are associated with repair paths.

A tiebreaking algorithm considers LFAs that satisfy certain conditions or have certain attributes. When there is more than one LFA, configure the **fast-reroute per-prefix** command with the **tie-break** keyword. If a rule eliminates all candidate LFAs, then the rule is skipped.

A primary path can have multiple LFAs. A routing protocol is required to implement default tiebreaking rules and to allow you to modify these rules. The objective of the tiebreaking algorithm is to eliminate multiple candidate LFAs, select one LFA per primary path per prefix, and distribute the traffic over multiple candidate LFAs when the primary path fails.

Tiebreaking rules cannot eliminate all candidates.

The following attributes are used for tiebreaking:

- Downstream—Eliminates candidates whose metric to the protected destination is lower than the metric of the protecting node to the destination.
- Linecard-disjoint—Eliminates candidates sharing the same linecard with the protected path.
- Shared Risk Link Group (SRLG)—Eliminates candidates that belong to one of the protected path SRLGs.
- Load-sharing—Distributes remaining candidates among prefixes sharing the protected path.
- Lowest-repair-path-metric—Eliminates candidates whose metric to the protected prefix is higher.
- Node protecting—Eliminates candidates that are not node protected.
- Primary-path—Eliminates candidates that are not ECMPs.
- Secondary-path—Eliminates candidates that are ECMPs.

Fast Reroute with Remote Loop-Free Alternate

Fast Reroute with Remote Loop-Free Alternate (FRR Remote LFA) feature enables you to tunnel a packet around a failed link to a remote loop-free alternate that is more than one hop away.

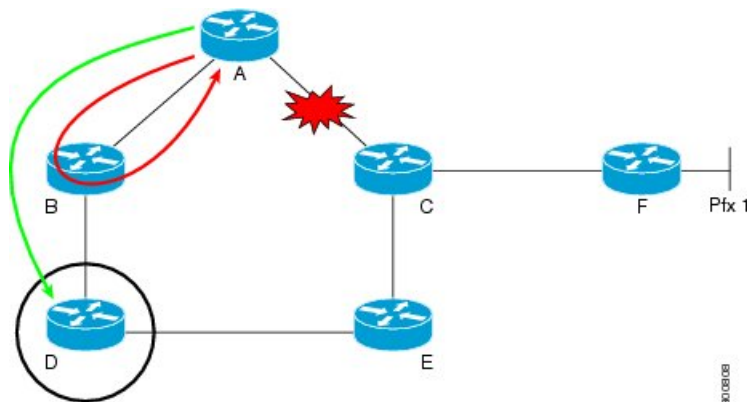
When a link or a router fails, distributed routing algorithms compute new routes that take into account the failure. The time taken for computation is called routing transition. Until the transition is complete and all routers are converged on a common view of the network, the connectivity between the source and destination pairs is interrupted. You can use the IP Loop-Free Alternate (LFA) Fast Reroute (FRR) to reduce the routing transition time to less than 50 milliseconds using a precomputed alternate next hop. When a router is notified of a link failure, the router immediately switches over to the repair path to reduce traffic loss. Note that the routing transition in IGP/BGP convergence can take up to several hundreds of milliseconds.

IP Loop-Free Alternate (LFA) Fast Reroute (FRR) supports the precomputation of repair paths. Intermediate System-to-Intermediate System (IS-IS) routing protocol enables the repair path computation. The resulting repair paths are sent to the Routing Information Base (RIB). Cisco Express Forwarding (formerly known as CEF) and Open Shortest Path First (OSPF) installs the repair path.

With IP local LFA FRR, IGP only compute directly connected neighbor as an LFA backup path to protect the given prefix's primary path. Label Distribution Protocol (LDP) sets up labeled backup LSP with the next-hop for the protected prefix. Some topologies (for example the commonly used ring-based topology) require protection that is not afforded by LFA FRR. In such cases, use the LDP-based FRR Remote LFA feature where IGP compute non-directly connected neighbor, which are more than one hop away, as LFA backup path to protect the given prefix's primary path. The LDP sets up labeled backup LSP with the remote next-hop for the protected prefix. LDP also sets up another transport LSP to tunnel traffic to remote next-hop without exposing the LFA backup label as learnt from remote node.

Consider the topology shown in the figure below:

Figure 11: FRR with Remote LFA with Ring Topology



Device A tries to send traffic destined to F to next-hop B. Device B cannot be used as an LFA for prefixes advertised by nodes C and F. The actual LFA is node D. However, node D is not directly connected to the protecting node A. To protect prefixes advertised by C, node A must tunnel the packet around the failed link A-C to node D, provided that the tunnel does not traverse the failing link.

FRR Remote LFA feature enables you to tunnel a packet around a failed link to a remote loop-free alternate that is more than one hop away. In the figure above, the green arrow between A and D shows the tunnel that is automatically created by the remote LFA feature to bypass looping.

Configuration

Configure remote FRR with remote LFA.

```

/* Configure FRR with remote LFA using IS-IS */
Router# configure
Router(config)# router isis ring
Router(config)# is-type level-1
Router(config-isis)# net 49.0001.0000.0000.0007.00
Router(config-isis)# nsf cisco
Router(config-isis)# address-family ipv4 unicast
Router(config-isis-af)# metric-style wide
Router(config-isis-af)# exit
Router(config-isis)# interface Loopback 0
Router(config-isis-if)# point-to-point
Router(config-isis-if)# address-family ipv4 unicast
Router(config-isis-if)# exit
Router(config-isis)# interface TenGigabitEthernet 0/0/0/4
Router(config-isis-if)# point-to-point
Router(config-isis-if)# address-family ipv4 unicast
/* To enable FRR LFA prefix dependent computation, configure the fast-reroute per-prefix
   command.
Or, to enable FRR LFA prefix independent per-link computation, configure the fast-reroute
   per-link command.*/
Router(config-isis-af)# fast-reroute per-prefix
Router(config-isis-af)# fast-reroute per-prefix remote-lfa tunnel mpls-ldp
Router(config-isis-af)# fast-reroute per-prefix remote-lfa prefix-list
Router(config-isis-af)# commit

/* Configure FRR with remote LFA using OSPF */
Router# configure
Router(config)# router ospf 50
Router(config-ospf)# router-id 10.1.1.1
Router(config-ospf)# address-family ipv4 unicast
Router(config-ospf-af)# area 0
Router(config-ospf-af)# interface Loopback 0
Router(config-ospf-af)# exit
Router(config-ospf)# interface HundredGigE0/2/0/0
/* To enable FRR LFA prefix dependent computation, configure the fast-reroute per-prefix
   command.
Or, to enable FRR LFA prefix independent per-link computation, configure the fast-reroute
   per-link command.*/
Router(config-isis-af)# fast-reroute per-prefix
Router(config-ospf-if)# fast-reroute per-prefix remote-lfa tunnel mpls-ldp
Router(config-ospf-if)# exit
Router(config-ospf)# exit

```

Running Configuration

Use either **fast-reroute per-prefix** or **fast-reroute per-link** command:

- To enable FRR LFA prefix dependent computation, configure the **fast-reroute per-prefix** command.
- To enable FRR LFA prefix independent per-link computation, configure the **fast-reroute per-link** command.

The following example shows IPv4 LFA FRR configurations. However, IPv6 LFA FRR is also supported.

The following is the FRR with remote LFA running configuration.

```

/* FRR with remote LFA with ISIS */
router isis ring
 is-type level-1
 net 49.0001.0000.0000.0007.00
 nsf cisco
 address-family ipv4 unicast
 fast-reroute per-prefix remote-lfa prefix-list abc
 metric-style wide

!
interface Loopback0
 point-to-point
 address-family ipv4 unicast
!
!
interface TenGigabitEthernet 0/0/0/4
 point-to-point
 address-family ipv4 unicast
 fast-reroute per-prefix remote-lfa prefix-list
 fast-reroute per-prefix remote-lfa tunnel mpls-ldp

/* FRR with remote LFA with OSPF */
router ospf 50
 router-id 10.1.1.1
 address-family ipv4 unicast
 area 0
 interface Loopback0
 !
 interface HundredGigE 0/2/0/0
 fast-reroute per-prefix remote-lfa tunnel mpls-ldp
 !
!

```

The following is the the FRR with local LFA running configuration.

```

/* FRR with local LFA with ISIS */
router isis ring
 is-type level-1
 net 49.0001.0000.0000.0007.00
 nsf cisco
 address-family ipv4 unicast
 metric-style wide
!
interface Loopback0
 point-to-point
 address-family ipv4 unicast
!
!
interface HundredGigE 0/2/0/0
 point-to-point
 address-family ipv4 unicast
 fast-reroute per-prefix

/* FRR with local LFA with OSPF */
router ospf 50
 router-id 10.1.1.1
 address-family ipv4 unicast
 area 0

```



```

interface Loopback0
!
interface HundredGigE 0/2/0/0
  fast-reroute per-prefix
!
!
!

```

Verification

The show outputs given in the following section display the details of the configuration of the FRR with LFA, and the status of their configuration.

```
/* Verify the route summary information about the specified routing table. */
```

```
RP/0//CPU0:router# show route 10.3.3.3
```

```

Routing entry for 10.3.3.3/32
  Known via "isis 44", distance 115, metric 20, type level-1
  Installed Nov 15 19:43:13.367 for 00:00:34
  Routing Descriptor Blocks
    10.1.1.1, from 10.3.3.3, via TenGigE0/0/0/0, Backup (remote)
      Remote LFA is 10.9.9.9
      Route metric is 0
    10.1.1.2, from 10.3.3.3, via TenGigE0/7/0/3, Protected
      Route metric is 20
  No advertising protos.

```

```
/* Verify the MPLS LDP configuration. */
```

```
RP/0//CPU0:router# show running mpls ldp
```

```
Codes:
```

```

- = GR label recovering, (!) = LFA FRR pure backup path
{} = Label stack with multi-line output for a routing path
G = GR, S = Stale, R = Remote LFA FRR backup

```

Prefix	Label In	Label(s) Out	Outgoing Interface	Next Hop	Flags
					G S R
192.0.2.0/24	16019	{ 16001 28006 }	Te0/0/0/0	10.1.1.1 (10.9.9.9)	(!) R
192.0.2.1/32	16013	ImpNull	Te0/7/0/3	192.0.2.1	
192.0.1.0/32	16014	{ 16001 16002 }	Te0/0/0/0	10.1.1.1 (10.9.9.9)	(!) R
		ImpNull	Te0/7/0/3	192.0.2.2	
10.9.9.9/32	16012	16001	Te0/0/0/0	10.1.1.1	
		28006	Te0/7/0/3	192.0.2.1	
10.23.1.0/24	16018	16004	Te0/0/0/0	10.1.1.1	(!)
		ImpNull	Te0/7/0/3	192.0.2.1	
10.34.1.0/24	16015	ImpNull	Te0/0/0/0	10.1.1.1	
10.0.0.1/32	16011	{ 16001 16013 }	Te0/0/0/0	10.1.1.1 (10.9.9.9)	(!) R
		16016	Te0/7/0/3	192.0.2.1	
10.100.0.2/32	16010	{ 16001	Te0/0/0/0	10.1.1.1	(!) R

Configuring IPv4 or IPv6 Loop-Free Alternate Fast Reroute Support: Example

The following example shows how to configure IPv4 LFA FRR. However, IPv6 LFA FRR is also supported.

```
router isis core
  is-type level-2-only
  net 47.0001.0000.0000.8888.00
  address-family ipv4 unicast
  metric-style wide
  exit
!
interface TenGigabitEthernet 0/0/0/4
  point-to-point
  address-family ipv4 unicast
    fast-reroute per-prefix
!
!

router ospf 50
  router-id 10.1.1.1
  address-family ipv4 unicast
  area 0
    interface Loopback0
    !
  interface HundredGigE 0/2/0/0
    fast-reroute per-prefix

!
!
```